ESCUELA POLITÉCNICA DE INGENIERÍA DE GIJÓN

MÁSTER UNIVERSITARIO EN INGENIERÍA DE TELECOMUNICACIÓN

ÁREA DE TEORÍA DE LA SEÑAL Y COMUNICACIONES

TRABAJO FIN DE MÁSTER Nº 201906

VISIBILITY AREAS FOR PHYSICAL AND VIRTUAL TRANSMITTERS IN MULTIPATH ASSISTED POSITIONING

D. JAIME SUÁREZ GARCÍA
TUTOR: D. JESÚS ALBERTO LÓPEZ FERNÁNDEZ
SUPERVISOR: D. MARKUS ULMSCHNEIDER

FECHA: Julio de 2019

Acronyms

AOA	angle of arrival
AWGN	additive white Gaussian noise
DLR	German Aerospace Center
DMC	dense multipath component
EKF	extended Kalman filter
GNSS	global navigation satellite system
IMU	inertial measurement unit
KEST	Kalman enhanced super-resolution tracking
KF	Kalman filter
LOS	line-of-sight
MAP	multipath assisted positioning
MC	Monte Carlo
MPC	multipath component
NLOS	non-line-of-sight
PDF	probability density function
PF	particle filter
RBPF	Rao-Blackwellized particle filter
SLAM	simultaneous localization and mapping
TOA	time of arrival

Contents

1.- Introduction

The present Master's thesis has been developed in a 6 month internship at the Mobile Radio Transmission group of the German Aerospace Center (DLR). It consists in the improvement of the Channel-SLAM positioning algorithm, which has been created by the already mentioned investigation group, by the estimation and usage of the visibility areas of transmitters.

In this chapter, the reader is introduced to the work through the scope definition and the explanation of the performed tasks. The motivation and objective of this thesis are specified in Section 1.1 and Section 1.2, respectively. The state of the art of the indoor positioning task is described in Section 1.3. The task schedule is included in Section 1.4, and the structure of the rest of the document is outlined in Section 1.5.

1.1.- Motivation

The usage of the location information has raised in the recent years in a broad range of applications. On the one hand, positioning represents a big business opportunity: it can be used to obtain information about the consumer's behavior in malls, cinemas or museums, among others, to improve the marketing strategy. Furthermore, it is needed in navigation, intelligent transportation systems, and it also has a big impact on the entertainment sector, e.g. in the improvement of augmented reality applications, or even in the social media. On the other hand, positioning is useful in emergency situations, as rescue and fire services. In this sense, the location of firefighters, trained dogs or policemen can provide serious benefits. Besides, medical personnel, patient and equipment tracking may also be important in hospitals.

Positioning in outdoor environments is generally carried out by means of satellite radio transmissions. However, the position accuracy of the global navigation satellite systems (GNSSs) gets drastically reduced in critical environments such as indoor scenarios or urban canyons [1]. In these scenarios, the non-line-of-sight (NLOS) condition and the multipath propagation cause the failure of the GNSSs.

On this matter, the Channel-SLAM algorithm is presented as a solution to carry out the location task in multipath environments using terrestrial signals of opportunity [2], [3], [4]. Rather than mitigating multipath, it relies on exploiting it: Channel-SLAM treats multipath components as signals emitted in a line-of-sight (LOS) condition from virtual transmitters. Hence, the multipath propagation increases the number of visible transmitters and improves the position accuracy. In fact, localization may be possible with only one physical transmitter.

In Channel-SLAM, the knowledge of the transmitters' position is crucial to perform the positioning task but, in general, such information is not available. Thus, every user uses the received signal components to jointly estimate its own location and the transmitters' position: simultaneous localization and mapping (SLAM) [5]. Each Channel-SLAM user stores a map formed by the estimated location of all the transmitters that have been seen throughout its path.

In multiple user scenarios, prior knowledge about the transmitters' locations can be obtained through map sharing [6]. Nevertheless, the Channel-SLAM algorithm is a relative positioning approach. This means that, unless any absolute reference to a global coordinate system is obtained, every user map is in a different local coordinate system. Thus, in order to use the information from other maps, transformation parameters, i.e. rotation and translation parameters, must be obtained. This process is known as map matching [7].

The Channel-SLAM map matching methods developed to date, so-called in this thesis *classical map matching methods*, are only based on the relative position of the transmitters. Depending on the scenario, ambiguities in the correspondences between transmitters from both maps may cause big errors in the map matching performance. It is expected that the inclusion of spatial information about the transmitters' visibility regions improves the robustness of the map matching process considerably. Thus, the motivation of the present Master's thesis lies in improving the Channel-SLAM performance through the improvement of the classical map matching methods.

1.2.- Objective

According to the motivation, the objective of the present Master's thesis resides in finding a way to use the transmitters' visibility regions to improve the performance of the classical map matching methods. As already stated, an improvement in those map matching methods inherently involves an improvement in the Channel-SLAM performance in multiple user scenarios, such that map crowd-sourcing can be successfully conducted.

For carrying out the described objective, a proper manner to store the transmitters' visibility information during the Channel-SLAM operation needs to be found. In addition, a suitable way to use the stored visibility information in the map matching process needs to be theoretically developed, and also implemented in a simulation software. Finally, the improvement over classical map matching methods must be evaluated through simulations.

Despite the proposed work with visibility areas is focused on improving the Channel-SLAM algorithm, it can be extrapolated to other positioning methods based on anchors or landmarks that are only visible in certain areas.

1.3.- State of the Art

The SLAM task in GNSS-denied environments, as indoor scenarios, has been a major challenge in the past decades. This problematic consists of localizing a certain user in a previously unexplored environment while incrementally constructing a consistent map of the scenario. Basically, the complexity of the presented problematic resides in the strong interdependence between the user localization and the mapping of the environment.

So far, a wide variety of solutions have been developed for carrying out the SLAM task. All SLAM schemes have in common that one or more users are localized while a map of the surroundings is estimated simultaneously. The assumed information source and how the map is conceived are clearly the two principal facts that make the operation of the different SLAM approaches to strongly differ.

Starting with the information source, an extended assumption is that information coming from sensors is available. There is a broad variety of sensors that may be used for SLAM. For example, the approach described in [8] assumes that user heading and speed information from an inertial measurement unit (IMU) is available, while the methods detailed in [9], [10] and [11] are based on sonar sensors. Visual information is used in other approaches: a single camera is used in the SLAM methods from [12], [13] and [14], and the information from multiple cameras is combined in the approach described in [15]. The authors in [16] consider a multi-sensor system, formed by a sonar sensor and a camera.

Other SLAM solutions rely on signals emitted from terrestrial radio transmitters. Such is the case of the approaches described in [17], [18] and [19], which assume the environment geometry and the physical transmitters' locations to be known in advance. Channel-SLAM [3] also relies on signals from terrestrial radio transmitters, but no prior information about the surroundings is needed to perform the algorithm. Other examples are the methods from [20] and [21], which are only based on the WiFi strength signal.

For its part, the map conception is precisely critical in the present Master's thesis. Several map definitions used in different SLAM approaches are described hereafter. Further information about SLAM maps can be found in [22].

- Grid maps

  This type of map tries to shape the physical environment by storing useful information in every cell of a grid. Commonly, the stored information represents an occupancy probability [23], but it can widely vary. Despite this map conception is the most popular in two-dimensional SLAM algorithms, it has problems with scalability.

- Feature maps

  This conception relies on storing the locations of specific landmarks used for positioning. The main advantage of feature maps is the scalability, while the major drawback corresponds to the data association: identification of landmarks over time.

- Semantic maps

  Every object of the environment is stored along with its characteristics and functionalities. In addition, relationships and interactions between objects are also stored. This abstract map conception is appropriate for goal-oriented behaviors and high-level reasoning.

- Hybrid maps

  Different map conceptions are combined and used simultaneously. A hybrid map increases the complexity of map storage and requires coordination between maps, but it can handle singular map inconsistencies.

Inherent in its operation, the Channel-SLAM algorithm stores the position of the different transmitters in a map. Thus, the Channel-SLAM map conception is consistent with the previously described feature map, where the transmitters are regarded as landmarks.

Accordingly, the Channel-SLAM map matching methods developed to date are only based on the relative position of the transmitters [7], since no more information is contained in the maps.

1.4.- Time Planning

The present Master's thesis has been started on the 1st of February 2019, while it has been completely finished by the 15th of July of the same year. Therefore, the work has been conducted in approximately 24 weeks.

The main tasks performed during this period are summarized in Table 1.1. Tasks are represented in chronological order, and they can be formed, in turn, by multiple sub-tasks. Task and sub-task durations are depicted in gray and light gray, respectively.

The first part of the accomplished work has been related to literature research, the middle part has been corresponded to theoretical and practical development, and the last part has focused mainly on evaluations and the thesis writing.

Task \ Week	2	4	6	8	10	12	14	16	18	20	22	24
Literature research	▓	▓	▓				▓					
Channel-SLAM		▓	▓	▓					▓	▓		
Implementation		░	░	░								
Simulation				░					░			
Obtainment of maps				░					░	░		
Visibility information					▓	▓	▓					
Storage: map estimation					░							
Visibility metric development					░	░						
Implementation						░	░					
Simulation							░					
Map matching						▓	▓	▓	▓	▓		
Standard method: research						░						
Rotation method: development							░					
Implementation							░	░	░			
Simulation									░	░		
Evaluation										▓	▓	
Writing							▓	▓	▓	▓	▓	▓

Table 1.1.- Time planning carried out during the development of the Master's thesis.

1.5.- Document Structure

The present document is formed by 5 chapters, where the first one is reserved for the introduction. The structure of the rest of the dissertation is depicted below.

Firstly, a Channel-SLAM overview is provided in Chapter 2. The main principles of the positioning algorithms based on the multipath propagation exploitation are also described. This is a theoretical chapter, where the relevant concepts for the understanding of the developed work are explained.

Secondly, how the visibility information of transmitters is stored, included in the Channel-SLAM algorithm, and used in the map matching methods is explained in Chapter 3. All the theoretical developed work is included in this chapter.

After that, the conducted work is critically evaluated by means of simulations in Chapter 4. The practical developed work, the simulated conditions and the main results are provided.

Lastly, a summary of the performed work, the principal outcomes derived from that work and a generic outlook are included in Chapter 5.

2.- Channel-SLAM

The Channel-SLAM algorithm is presented as a solution for the user localization task in multipath scenarios, such as urban canyons or indoors. This approach relies on exploiting multipath propagation, instead of mitigating it, to obtain the location of the user and to create a map of transmitters simultaneously. In addition, no prior knowledge is required to perform the algorithm.

Basically, the Channel-SLAM structure is composed of two stages. In the first stage, the parameters of every received signal component are estimated and tracked over time. In the second stage, the resulting estimates from the previous step are used to jointly locate the user and the transmitters: SLAM.

In this chapter, the reader will find the key concept of Channel-SLAM. The main principles that determine Channel-SLAM are introduced in Section 2.1, while the Channel-SLAM structure and algorithm are detailed in Section 2.2.

2.1.- Principles of Channel-SLAM

Three crucial concepts for the Channel-SLAM understanding are explained in this section. A different subsection is dedicated to each principle.

2.1.1.- Multipath Assisted Positioning

In a multipath environment, every user receives a set of signal components, most of them after being reflected at walls, or diffracted at point scatterers. These reflected and diffracted signals are called multipath components (MPCs). The Channel-SLAM algorithm uses MPCs as well as LOS signals to locate the user and the transmitters.

The fundamental idea of multipath assisted positioning (MAP), adopted in Channel-SLAM, is to regard every received signal as being sent by a transmitter in a LOS condition. This means that no distinction among MPCs and LOS signal components is made. Therefore, every received signal has an associated transmitter, which can be physical or virtual depending on the received signal: LOS or MPC, respectively.

The position of the virtual transmitters depends on the structure of the scenario and the position of the physical transmitters. However, as long as the physical transmitters and the scenario remain static over time, the virtual transmitters also remain static, as it can be seen in Figures 2.1 and 2.2. For SLAM, time of arrival (TOA) and angle of arrival (AOA) estimates of every received signal may be used.

Virtual transmitters related to reflections, as in Figure 2.1, are inherently time synchronized to the physical transmitter: the traveled distance of the signal component that is reflected in the wall is exactly the same as the distance between the user and the position of the virtual transmitter. Therefore, it will be possible to use the LOS signal of the virtual transmitter for localization.

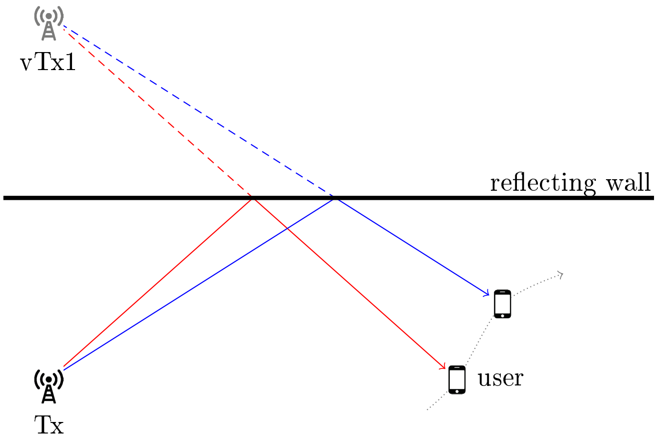


Figure 2.1.- The signal from the physical transmitter Tx, after being reflected at the wall, is received at the user and interpreted as a LOS signal from the virtual transmitter vTx1. The position of vTx1 is the location of Tx mirrored at the reflecting wall, and it remains static as long as the physical transmitter and the reflecting wall remain static.

Virtual transmitters related to point scatterers, as in Figure 2.2, have an additional propagation delay τ_0 which is proportional to the distance between the point scatterer itself and the physical transmitter. This propagation delay has to be taken into account to properly locate the virtual transmitter position. It can be interpreted as a clock offset of the virtual transmitter.

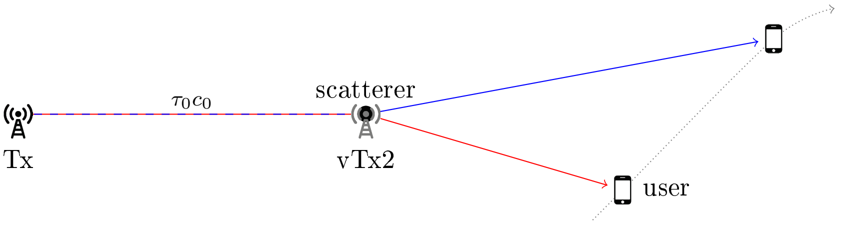


Figure 2.2.- The signal from the physical transmitter Tx, after being scattered, is received at the user and interpreted as a LOS signal from the virtual transmitter vTx2. The virtual transmitter vTx2 has an additional clock offset τ_0, which is the distance between the transmitters divided by the speed of light c_0.

The described concept of single reflections and scatterings can be generalized to the case of multiple reflections and scatterings or a combination of them by applying the same reasoning. If the signal undergoes only reflections, the corresponding virtual transmitter is inherently time synchronized to the physical transmitter. In case the signal is scattered at least once, then the pertinent virtual transmitter has a clock offset τ_0.

Finally, assuming that the transmitters' locations are known, at least three of them are needed to avoid ambiguities and calculate the user location in a two-dimensional scenario. In a multipath scenario, every single physical transmitter will generate multiple virtual transmitters. Therefore, the user localization can be performed in a multipath environment with only one physical transmitter thanks to MAP and the multipath components exploitation.

2.1.2.- Recursive Bayesian Estimation

Recursive Bayesian estimation [24] is the basis of the Channel-SLAM position estimation. The objective of this method is to recursively estimate the evolution of a state vector $\boldsymbol{x}$ over time. The state vector contains all relevant information required to describe the system under investigation. For example, in tracking problems, the coordinates of the target could form the state vector.

Estimation approaches require measurements related to the state vector. These measurements, or observations, are typically subject to measurement noise and represented by the measurement vector $\boldsymbol{z}$. Following the tracking example, this measurement vector could be formed by TOA, AOA, or signal received power measurements, among others.

In order to apply the recursive Bayesian estimation in a dynamic system, two models are required. Firstly, the system model, which describes the evolution of the state with time. The system model can be also found in the literature as process or evolution model. Secondly, the measurement model, which relates the noisy observations to the state.

Then, the evolution of the state vector $\boldsymbol{x}$ is modeled as

$$\boldsymbol{x}_k = \boldsymbol{f}_k\left(\boldsymbol{x}_{k-1}, \boldsymbol{v}_{k-1}\right), \tag{2.1}$$

where the state evolution function $\boldsymbol{f}(\cdot)$ is assumed to be known, the index k denotes the time instant, and $\boldsymbol{v}_{k-1}$ denotes a sample of the system noise. The state is related to the measurement $\boldsymbol{z}_k$ as

$$\boldsymbol{z}_k = \boldsymbol{h}_k\left(\boldsymbol{x}_k, \boldsymbol{n}_k\right), \tag{2.2}$$

where the measurement model $\boldsymbol{h}(\cdot)$ is again a known function and $\boldsymbol{n}_k$ is a sample of the measurement noise.

Recursive Bayesian estimation works in two stages that are repeated over time: prediction and update. First, the prediction stage uses the system model to predict the state probability density function (PDF) from one time instant to the next. Then, the update stage uses the Bayes' theorem to update the state PDF with the latest measurement.

In this way, the corresponding PDFs can be calculated recursively by

$$\mathrm{p}\left(\boldsymbol{x}_k|\boldsymbol{z}_{1:k-1}\right) = \int \mathrm{p}\left(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}\right) \mathrm{p}\left(\boldsymbol{x}_{k-1}|\boldsymbol{z}_{1:k-1}\right) \mathrm{d}\boldsymbol{x}_{k-1} \tag{2.3}$$

for the prediction step, and by

$$\mathrm{p}\left(\boldsymbol{x}_k|\boldsymbol{z}_{1:k}\right) = \frac{1}{c_k} \mathrm{p}\left(\boldsymbol{z}_k|\boldsymbol{x}_k\right) \mathrm{p}\left(\boldsymbol{x}_k|\boldsymbol{z}_{1:k-1}\right) \tag{2.4}$$

for the update step, where c_k is a normalizing constant and $\boldsymbol{z}_{1:k}$ denotes the measurements from time instant 1 to k. The state transition prior $\mathrm{p}\left(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}\right)$ and the measurement likelihood $\mathrm{p}\left(\boldsymbol{z}_k|\boldsymbol{x}_k\right)$ are obtained from the system model in Equation (2.1) and the measurement model in Equation (2.2), respectively.

It should be noted that the state PDF $\mathrm{p}\left(\boldsymbol{x}_{k-1}|\boldsymbol{z}_{1:k-1}\right)$ is needed to calculate its value at next time instant: $\mathrm{p}\left(\boldsymbol{x}_k|\boldsymbol{z}_{1:k}\right)$. Thus, recursion is the key of the presented estimation approach.

The recurrence of Equations (2.3) and (2.4) form the optimal Bayesian solution. However, in practice, this recursive propagation can only be exactly determined in a specified set of cases. Further information about recursive Bayesian optimal and suboptimal algorithms, and its constraints, can be found in [25].

2.1.3.- Particle Filter

In many situations, an optimal solution to recursive Bayesian estimation cannot be obtained since the integral shown in Equation (2.3) cannot be expressed in closed form, or its calculation becomes intractable. The proposed tracking problem is one of those situations: the measurement model of the present problem, where the AOA and TOA are measured, is non-linear. This means that the Kalman filter (KF), an optimal algorithm, cannot be used. In fact, due to the high non-linearity of the present measurement model, the extended Kalman filter (EKF), which works with local linearizations, shows a bad performance.

In light of this, the particle filter (PF) [26], also known as Bayesian bootstrap, presents itself as a non-optimal but in most cases well-performing solution to be used for recursive Bayesian estimation, and thus in the second stage of the Channel-SLAM algorithm. The great strength of this technique is that no restrictions are placed on the system and measurement model, nor on the system and measurement noise.

The PF is a technique for implementing a recursive Bayesian filter using Monte Carlo (MC) simulations. The fundamental idea is to represent the PDF shown in Equation (2.4) by a set of random samples from the state space, called particles, with associated weights. As the number of particles grows, this MC characterization becomes a proper representation of the PDF, and the method approaches to an optimal algorithm.

The idea expressed in the previous paragraph can be summarized in

$$\mathrm{p}\left(\boldsymbol{x}_k|\boldsymbol{z}_{1:k}\right) \approx \sum_{i=1}^{N_\mathrm{p}} w_k^{<i>}\delta\big(\boldsymbol{x}_k - \boldsymbol{x}_k^{<i>}\big), \tag{2.5}$$

where $\delta(\cdot)$ is the Dirac delta function, N_p is the number of particles, $\boldsymbol{x}_k^{<i>}$ denotes the i^{th} particle, and $w_k^{<i>}$ denotes the i^{th} particle's associated weight. The weights are normalized such that

$$\sum_{i=1}^{N_\mathrm{p}} w_k^{<i>} = 1. \tag{2.6}$$

Following the recursive Bayesian estimation approach, the PF method also uses the prediction and update stages. Firstly, in the prediction step, new samples are drawn in accordance with the system model. Then, in the update step, the weights associated to the particles are updated through the likelihood function $\mathrm{p}\left(\boldsymbol{z}_k|\boldsymbol{x}_k\right)$, which is defined by the measurement model. Finally, weights are normalized. The higher the weight, the more reliable the associated particle.

Equation (2.7) shows the i^{th} particle weight update.

$$w_k^{<i>} \propto w_{k-1}^{<i>}\mathrm{p}\left(\boldsymbol{z}_k|\boldsymbol{x}_k^{<i>}\right) \tag{2.7}$$

Further information about the implementation of the system and measurement models in Channel-SLAM is contained in Section 2.2.2, once all the parameters of the Channel-SLAM algorithm are introduced.

Continuing with the PF technique, an inherent problem to this method is the degeneracy phenomenon: after some iterations, most particles will have negligible weight. This means that most part of the computational cost of the algorithm is dedicated to processing particles that have an insignificant contribution to the approximation of the posterior PDF and, therefore, to the algorithm performance.

A suitable measure of the degeneracy problem, called effective sample size (N_{eff}), is obtained in [27]. However, the effective sample size cannot be evaluated exactly. Therefore, the following effective sample size estimate $\widehat{N}_{eff}$ is commonly used to measure the degeneracy problem:

$$\widehat{N}_{eff} = \frac{1}{\sum\limits_{i=1}^{N_\mathrm{p}} \left(w_k^{<i>}\right)^2}. \tag{2.8}$$

An extended and simple solution to the degeneracy problem is the application of the so-called resampling algorithm [26] whenever a significant degeneracy is observed, i.e. when $\widehat{N}_{eff}$ falls below a threshold. After its application, a new set of N_p particles obtained from the approximate discrete representation of the posterior PDF $\mathrm{p}\left(\boldsymbol{x}_k|\boldsymbol{z}_{1:k}\right)$, given by Equation (2.5), replaces old particles.

The key idea of the resampling algorithm is to delete particles with negligible weight, replace them by existing particles with larger weight, and reset weights to $w_k^{<i>} = 1/N_\mathrm{p}$. A pseudo-code description of this algorithm can be found in [25].

2.2.- The Channel-SLAM Derivation

In this section, a simplified version of the Channel-SLAM algorithm from [2], [3] and [4] is described. Despite no prior knowledge is required to perform the Channel-SLAM algorithm, it is assumed in the present approach that information coming from an IMU is available.

This section is structured as follows. Firstly, some Channel-SLAM parameters are defined and the algorithm structure is specified in Subsection 2.2.1. Then, the second stage of the algorithm, strongly related with the new ideas developed in the present Master's thesis, is discussed in more depth in Subsection 2.2.2. Lastly, the algorithm initialization is detailed in Subsection 2.2.3, and some details about its implementation are depicted.

2.2.1.- Structure of Channel-SLAM

Before describing the structure of Channel-SLAM, the signal model is specified. The wireless radio frequency propagation channel is assumed to be a linear and time-variant multipath channel. Thus, the received signal $r\left(t\right)$ is regarded as a superposition of signal components of the broadcasted signal $s\left(t\right)$. At time k, the j^{th} signal component is defined by a delay and a complex amplitude: $d_j\left(k\right)$ and $a_j\left(k\right)$, respectively. Therefore, the received signal at time instant k can be written as

$$r\left(\tau, k\right) = \sum_j a_j\left(k\right) s\left(\tau - d_j\left(k\right)\right) + n_s\left(\tau\right), \tag{2.9}$$

where $n_s\left(\tau\right)$ is a sample from a colored noise sequence. This noise includes both dense multipath components (DMCs), i.e. components that cannot be characterized by a superposition of plane waves, and additive white Gaussian noise (AWGN). Moreover, the channel is assumed to remain constant during the sample intervals at every time instant k.

As already stated in the beginning of the present chapter, the Channel-SLAM algorithm is made up of two stages: the parameter estimation and the position estimation. The objective of the first stage is to obtain AOA and TOA estimates of every signal component from the received signal $r\left(t\right)$. In order to carry out those estimates, the Kalman enhanced super-resolution tracking (KEST) algorithm described in [28] is used. In [29], the KEST algorithm has been proved to be a proper solution to accomplish this objective, being more robust in solving close signal components than the EKF method.

Thus, in the Channel-SLAM algorithm, the measurement vector $\boldsymbol{z}_k$ is defined as

$$\boldsymbol{z}_k = \left[d_k^{<1>} \ ... \ d_k^{<N_{\mathrm{TX}}>} \quad \theta_k^{<1>} \ ... \ \theta_k^{<N_{\mathrm{TX}}>} \right]^T , \tag{2.10}$$

where $d_k^{<j>}$ and $\theta_k^{<j>}$ denote the j^{th} signal component TOA and AOA estimates, respectively, at time instant k. It should be noted than the number of transmitters N_{TX}, i.e. the number of received signal components, is time variant, but the time instant k is omitted for notational brevity.

The objective of the second stage of the Channel-SLAM algorithm is to jointly determine the position of the user and the transmitters. In order to achieve that objective, the measurements $\boldsymbol{z}_k$ taken at each time instant along with information resulting from an IMU are used.

Therefore, the user state vector at time instant k, $\boldsymbol{x}_{\mathrm{u},k}$, is formed by the two-dimensional user coordinates and velocity components, as follows:

$$\boldsymbol{x}_{\mathrm{u},k} = \left[x_k \quad y_k \quad v_{x,k} \quad v_{y,k} \right]^T . \tag{2.11}$$

Regarding transmitters, as stated in Subsection 2.1.1, it is necessary to determine their clock offset τ_0 to properly estimate their position. Furthermore, they are assumed to be static. Thus, the state vector of the j^{th} transmitter at time instant k, $\boldsymbol{x}_{\mathrm{TX},k}^{<j>}$, is composed of the two-dimensional transmitter coordinates and its clock offset:

$$\boldsymbol{x}_{\mathrm{TX},k}^{<j>} = \left[x_{\mathrm{TX},k}^{<j>} \quad y_{\mathrm{TX},k}^{<j>} \quad \tau_{0,k}^{<j>} \right]^T . \tag{2.12}$$

Finally, the entire state vector at time instant k is formed by stacking the user state vector and the N_{TX} transmitters' state vector:

$$\boldsymbol{x}_k = \left[\boldsymbol{x}_{\mathrm{u},k}{}^T \quad \boldsymbol{x}_{\mathrm{TX},k}^{<1>}{}^T \ ... \ \boldsymbol{x}_{\mathrm{TX},k}^{<N_{\mathrm{TX}}>T} \right]^T . \tag{2.13}$$

The Channel-SLAM structure is summarized in Figure 2.3. It should be underlined that no prior knowledge about the scenario is needed to perform the Channel-SLAM algorithm.

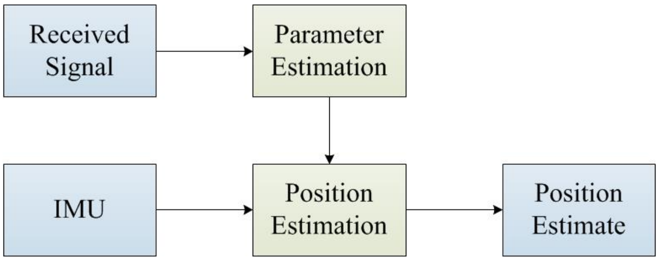


Figure 2.3.- Channel-SLAM stages: parameter and position estimation.

2.2.2.- Position Estimation

As already stated, the main goal of the position estimation stage is to simultaneously locate the user and the physical and virtual transmitters over time. In order to carry out this objective, the PF, a recursive Bayesian estimator with few constraints on the underlying filtering problem, is used to determine the evolution of the entire state vector $\boldsymbol{x}_k$.

The state vector $\boldsymbol{x}_k$ shown in Equation (2.13) presents a large number of dimensions, which can be a problem. The first step to reduce the number of dimensions is to separate the transmitters' state space from the user state space. In this way, the posterior PDF $\mathrm{p}\left(\boldsymbol{x}_k|\boldsymbol{z}_{1:k}\right)$ can be factorized as

$$\mathrm{p}\left(\boldsymbol{x}_k|\boldsymbol{z}_{1:k}\right) = \mathrm{p}\left(\boldsymbol{x}_{\mathrm{TX},k}, \boldsymbol{x}_{\mathrm{u},k}|\boldsymbol{z}_{1:k}\right) = \mathrm{p}\left(\boldsymbol{x}_{\mathrm{u},k}|\boldsymbol{z}_{1:k}\right)\mathrm{p}\left(\boldsymbol{x}_{\mathrm{TX},k}|\boldsymbol{x}_{\mathrm{u},k}, \boldsymbol{z}_{1:k}\right). \tag{2.14}$$

In addition, signal components arriving at the receiver are assumed to be independent from each other, because MPCs are assumed to interact with distinct objects. Therefore, independence among measurements for different transmitters can also be assumed, and the following factorization can be made:

$$\mathrm{p}\left(\boldsymbol{x}_{\mathrm{TX},k}|\boldsymbol{x}_{\mathrm{u},k}, \boldsymbol{z}_{1:k}\right) = \prod_{j=1}^{N_{\mathrm{TX}}} \mathrm{p}\left(\boldsymbol{x}_{\mathrm{TX},k}^{<j>}|\boldsymbol{x}_{\mathrm{u},k}, \boldsymbol{z}_{1:k}^{<j>}\right). \tag{2.15}$$

With this, the posterior PDF shown in Equation (2.14) can be rewritten as

$$\mathrm{p}\left(\boldsymbol{x}_k|\boldsymbol{z}_{1:k}\right) = \mathrm{p}\left(\boldsymbol{x}_{\mathrm{u},k}|\boldsymbol{z}_{1:k}\right)\prod_{j=1}^{N_{\mathrm{TX}}} \mathrm{p}\left(\boldsymbol{x}_{\mathrm{TX},k}^{<j>}|\boldsymbol{x}_{\mathrm{u},k}, \boldsymbol{z}_{1:k}^{<j>}\right). \tag{2.16}$$

The previous equation shows that it is possible to apply individual recursive Bayesian estimators to solve the present SLAM problem. Thus, individual system and measurement models are required for the user and the transmitters.

As the scenario and the physical transmitters are assumed to be static, the virtual transmitters are static as well. Therefore, the transmitter system model $\boldsymbol{f}_{\mathrm{TX},k}\left(\cdot\right)$ can be expressed as

$$\boldsymbol{f}_{\mathrm{TX},k}\left(\boldsymbol{x}_{\mathrm{TX},k-1}^{<j>}, \boldsymbol{v}_{\mathrm{TX},k-1}\right) = \boldsymbol{x}_{\mathrm{TX},k-1}^{<j>} + \boldsymbol{v}_{\mathrm{TX},k-1}, \tag{2.17}$$

where $\boldsymbol{v}_{\mathrm{TX},k}$ is a vector formed by samples of the transmitter system noise at time instant k. As the transmitter position is expected to be static, a low value of the transmitter system noise variance should be selected.

Incorporating sensor data from an IMU, the user system model $\boldsymbol{f}_{\mathrm{u},k}\left(\cdot\right)$ can be defined as

$$\boldsymbol{f}_{\mathrm{u},k}\left(\boldsymbol{x}_{\mathrm{u},k-1}, \boldsymbol{v}_{\mathrm{u},k-1}\right) = \boldsymbol{F}_{\mathrm{u},k}\,\tilde{\boldsymbol{x}}_{\mathrm{u},k-1} + \boldsymbol{v}_{\mathrm{u},k-1}, \tag{2.18}$$

where $\boldsymbol{v}_{\mathrm{u},k}$ is a vector formed by samples of the user system noise at time instant k, $\tilde{\boldsymbol{x}}_{\mathrm{u},k-1}$ is the user state vector at time instant $k-1$ with updated velocity information from time instant k

$$\tilde{\boldsymbol{x}}_{\mathrm{u},k-1} = \begin{pmatrix} x_{k-1} \\ y_{k-1} \\ v_{x,k} \\ v_{y,k} \end{pmatrix}, \tag{2.19}$$

and $\boldsymbol{F}_{\mathrm{u},k}$ is the user system model matrix defined as follows, where Δ_t is the time difference between adjacent time instants:

$$\boldsymbol{F}_{\mathrm{u},k} = \begin{pmatrix} 1 & 0 & \Delta_t & 0 \\ 0 & 1 & 0 & \Delta_t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{2.20}$$

Basically, the product $\boldsymbol{F}_{\mathrm{u},k}\,\tilde{\boldsymbol{x}}_{\mathrm{u},k-1}$ updates the user state vector to time instant k using the velocity obtained from the IMU:

$$\boldsymbol{F}_{\mathrm{u},k}\;\tilde{\boldsymbol{x}}_{\mathrm{u},k-1} = \begin{pmatrix} 1 & 0 & \Delta_t & 0 \\ 0 & 1 & 0 & \Delta_t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_{k-1} \\ y_{k-1} \\ v_{x,k} \\ v_{y,k} \end{pmatrix} = \begin{pmatrix} x_{k-1} + v_{x,k}\Delta_t \\ y_{k-1} + v_{y,k}\Delta_t \\ v_{x,k} \\ v_{y,k} \end{pmatrix} = \begin{pmatrix} x_k \\ y_k \\ v_{x,k} \\ v_{y,k} \end{pmatrix}. \tag{2.21}$$

Depending on the available information, the previous definitions may change. As an example, only heading change measurements from gyroscope are assumed to be available in [30].

Regarding the measurement model, as mentioned, the components that form the measurement vector $\boldsymbol{z}_k$ are assumed to be independent. Moreover, the TOA and the AOA measurement noise for the j^{th} transmitter are modeled as AWGN with variances $\sigma^2_{d,j}$ and $\sigma^2_{\theta,j}$, respectively. Thus, the likelihood function $\mathrm{p}\left(\boldsymbol{z}_k|\boldsymbol{x}_k\right)$ can be expressed as

$$\mathrm{p}\left(\boldsymbol{z}_k|\boldsymbol{x}_k\right) = \prod_{j=1}^{N_{\mathrm{TX}}} \frac{1}{\sqrt{2\pi}\sigma_{d,j}}\, e^{-\frac{\left(\hat{d}_k^{<j>} - d_k^{<j>}\right)^2}{2\,\sigma^2_{d,j}}}\; \frac{1}{\sqrt{2\pi}\sigma_{\theta,j}}\, e^{-\frac{\left(\hat{\theta}_k^{<j>} - \theta_k^{<j>}\right)^2}{2\,\sigma^2_{\theta,j}}}, \tag{2.22}$$

where $d_k^{<j>}$ and $\theta_k^{<j>}$ are the estimates obtained by KEST from Equation (2.10), the predicted TOA $\hat{d}_k^{<j>}$ between the user and the j^{th} transmitter at time instant k is calculated as

$$\hat{d}_k^{<j>} = \frac{1}{c_0}\left\|\left[x_k \quad y_k\right]^T - \left[x_{\mathrm{TX},k}^{<j>} \quad y_{\mathrm{TX},k}^{<j>}\right]^T\right\| + \tau_{0,k}^{<j>}, \tag{2.23}$$

where $\|\cdot\|$ is the Euclidean norm of a vector, and the predicted AOA $\hat{\theta}_k^{<j>}$ is given by

$$\hat{\theta}_k^{<j>} = \mathrm{atan2}\left(y_{\mathrm{TX},k}^{<j>} - y_k, x_{\mathrm{TX},k}^{<j>} - x_k\right) - \mathrm{atan2}\left(v_{y,k}, v_{x,k}\right). \tag{2.24}$$

The function atan2 (y, x) returns the counter-clockwise angle between the x-axis and the line connecting the origin with the point (x, y). Furthermore, the second term of the Equation (2.24) is included because the AOA KEST estimates are referred to the user movement direction.

With all the models being defined, the recursive Bayesian method used in the position estimation stage of the Channel-SLAM algorithm is specified below. As already discussed, the factorization shown in Equation (2.16) allows for estimating the user state $\boldsymbol{x}_{\mathrm{u},k}$ and each transmitter state $\boldsymbol{x}_{\mathrm{TX},k}^{<j>}$ independently. This fact leads to the Rao-Blackwellized particle filter (RBPF) ([30], [31], [32]), a method that uses multiple PFs to determine the evolution of each state vector.

In particular, the Channel-SLAM RBPF technique is based on one PF to estimate the user position, and multiple subordinate PFs (subPFs) to estimate the position of the transmitters. The key idea is that every user particle independently estimates the position of the transmitters using N_{TX} subPFs. In other words, every user particle independently estimates its own environment map.

Therefore, the RBPF method applied in Channel-SLAM uses subPFs inside the user PF to recursively estimate the transmitters' state vectors $\boldsymbol{x}_{\mathrm{TX},k}^{<j>}$ based on the user state, as expressed in Equation (2.15). Simultaneously, the user PF estimates the evolution of the user state vector $\boldsymbol{x}_{\mathrm{u},k}$.

It should be noted that the RBPF formulation allows to utilize a different number of particles in each transmitter PF. However, for notational brevity, the transmitter index j is omitted in $N_{\mathrm{p,TX}}$. The number of particles of the user PF is denoted by $N_{\mathrm{p,u}}$.

Following Equation (2.5), the ideas expressed in previous paragraphs can be summarized as follows. The user posterior PDF is expressed as

$$\mathrm{p}\left(\boldsymbol{x}_{\mathrm{u},k} | \boldsymbol{z}_{1:k}\right) = \sum_{i=1}^{N_{\mathrm{p,u}}} w_k^{<i>} \delta\big(\boldsymbol{x}_{\mathrm{u},k} - \boldsymbol{x}_{\mathrm{u},k}^{<i>}\big), \tag{2.25}$$

where $\boldsymbol{x}_{\mathrm{u},k}^{<i>}$ is the i^{th} user particle and $w_k^{<i>}$ its associated weight. Besides, the posterior PDF of $\boldsymbol{x}_{\mathrm{TX},k}^{<i,j>}$, i.e. the j^{th} transmitter for the i^{th} user particle, is represented as

$$\mathrm{p}\left(\boldsymbol{x}_{\mathrm{TX},k}^{<i,j>} | \boldsymbol{x}_{\mathrm{u},k}^{<i>}, \boldsymbol{z}_{1:k}\right) = \sum_{l=1}^{N_{\mathrm{p,TX}}} w_k^{<i,j,l>} \delta\Big(\boldsymbol{x}_{\mathrm{TX},k}^{<i,j>} - \boldsymbol{x}_{\mathrm{TX},k}^{<i,j,l>}\Big), \tag{2.26}$$

where $\boldsymbol{x}_{\mathrm{TX},k}^{<i,j,l>}$ is the l^{th} particle of the j^{th} transmitter for the i^{th} user particle, and $w_k^{<i,j,l>}$ its associated weight.

Furthermore, as every PF approach, the RBPF technique recursively applies the prediction and the update stages. First, in the prediction stage, the user system model depicted in Equation (2.18) is applied on every user particle. Then, the transmitter system model described in Equation (2.17) is employed in every transmitter particle.

Afterward, in the update stage, the weights of all transmitter particles are updated. In the next step the user particles are updated and, lastly, all particle weights are normalized. The remainder of the present subsection is dedicated to describe the Channel-SLAM RBPF particle updates.

Starting with the transmitters, the l^{th} transmitter particle weight $w_k^{<i,j,l>}$ of the j^{th} transmitter for the i^{th} user particle is updated by

$$w_k^{<i,j,l>} \propto w_{k-1}^{<i,j,l>} \mathrm{p}\left(\boldsymbol{z}_k | \boldsymbol{x}_{\mathrm{u},k}^{<i>}, \boldsymbol{x}_{\mathrm{TX},k}^{<i,j,l>}\right), \tag{2.27}$$

where the likelihood function is analog to the one shown in Equation (2.22), but the estimates from KEST for the j^{th} transmitter are now compared with the predicted TOA and AOA measurements between the l^{th} transmitter particle of the j^{th} transmitter, and the i^{th} user particle:

$$\hat{d}_k^{<i,j,l>} = \frac{1}{c_0} \left\| \left[x_k^{<i>} \quad y_k^{<i>}\right]^T - \left[x_{\mathrm{TX},k}^{<i,j,l>} \quad y_{\mathrm{TX},k}^{<i,j,l>}\right]^T \right\| + \tau_{0,k}^{<i,j,l>}, \tag{2.28}$$

$$\hat{\theta}_k^{<i,j,l>} = \mathrm{atan2}\left(y_{\mathrm{TX},k}^{<i,j,l>} - y_k^{<i>}, x_{\mathrm{TX},k}^{<i,j,l>} - x_k^{<i>}\right) - \mathrm{atan2}\left(v_{y,k}^{<i>}, v_{x,k}^{<i>}\right). \tag{2.29}$$

Finally, once all the transmitter particle weights have been updated, the user particle weights are updated as

$$w_k^{<i>} \propto w_{k-1}^{<i>} \prod_{j=0}^{N_{\mathrm{TX}}} \sum_{l=1}^{N_{\mathrm{p,TX}}} w_k^{<i,j,l>}, \tag{2.30}$$

and all transmitter and user particle weights are normalized. A complete description of the Channel-SLAM Rao-Blackwellization, and a full derivation of Equations (2.25), (2.26), (2.27) and (2.30) can be found in [4] and [30].

2.2.3.- Initialization and Implementation

In the previous subsection, the Channel-SLAM recursive Bayesian estimation method (RBPF) has been explained. However, that recursive model needs some start points, i.e. the state vectors need to be initialized.

For the user state vector $\boldsymbol{x}_{\mathrm{u},k}$, there are two options. If the initial user position and heading are known as a prior, all user particles can be initialized accordingly and Channel-SLAM becomes an absolute positioning algorithm. Alternatively, if the initial user position in unknown, all user particles are initialized to the same random coordinates, and Channel-SLAM becomes a relative positioning algorithm.

When the KEST algorithm detects a new signal component, it also obtains TOA and AOA measurements. Therefore, the Channel-SLAM algorithm is capable of using those measurements to initialize the corresponding transmitter particles. Nevertheless, the KEST algorithm cannot detect if the received signal component has been scattered

$(\tau_0^{<j>} \neq 0)$, or if it comes from a physical transmitter or a reflection $(\tau_0^{<j>} = 0)$. Thus, there is a distance ambiguity that can only be solved with measurements at future time instants.

In this way, every time a new signal component is detected, the corresponding transmitter particles are initialized such that they occupy a large area in the measured AOA direction. After some iterations, the position of the particles will converge to the transmitter position.

A high number of transmitter particles should be employed to occupy a large area. However, it is a waste of resources to continue using that amount of particles after some iterations, because the uncertainty of the transmitter state tends to decrease. It is thus essential to apply the so-called grid resampling algorithm to adapt the number of particles to the uncertainty of the transmitter estimate. This algorithm is basically the one depicted at the end of Subsection 2.1.3, now with a space constraint: only a certain number of particles are allowed to exist in a specified area. Thus, the number of particles will decrease over time, and with it the occupied memory and the computational complexity.

A transmitter initialization is outlined in Figure 2.4. The user trajectory is represented in blue, and the user particles are drawn in orange. Furthermore, the transmitter particles of vTx1, vTx2 and vTx3 for a single user particle are also drawn.

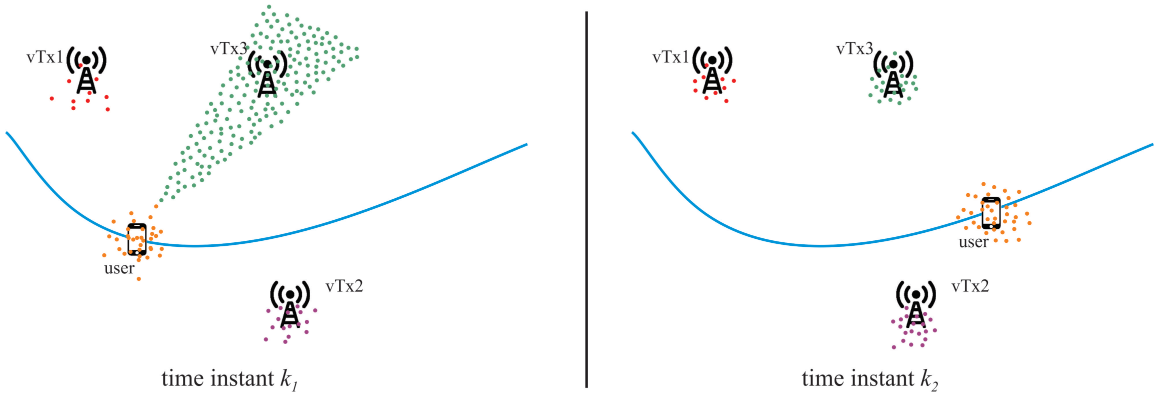


Figure 2.4.- Transmitter initialization outline. The virtual transmitter vTx3 is detected at time instant k_1. After some iterations, the transmitter particles of vTx3 concentrate around the transmitter position at time instant k_2. All the transmitter particles represented in this scheme belong to a single user particle; every user particle makes its own estimation of the transmitters' position.

The effect of the grid resampling algorithm can be appreciated in the previous illustration. A large number on transmitter particles is used when the uncertainty of the transmitter position estimate is high, as happens with the virtual transmitter vTx3 at time instant k_1. Nevertheless, when the uncertainty of the transmitter position gets reduced, the number of transmitter particles decrease, as it can be appreciated at time instant k_2.

As stated in Subsection 2.2.2, every user particle independently estimates the transmitters' position in the Channel-SLAM algorithm. This means that $N_{\mathrm{p,u}}$ maps are generated. In order to combine them, all the particles of each transmitter from all user particles estimates are merged taking the user particle weights into account. Finally, the grid resampling algorithm is applied to reduce the number of particles. The described process in which a unique map is obtained through the combination of the user particle maps is called map merging.

3.- Visibility Information

As already stated in Section 1.3, the Channel-SLAM map is a feature map where physical and virtual transmitters are used as landmarks. Thus, the Channel-SLAM map matching methods developed to date are based on the relative locations of the transmitters. Such approaches may fail when ambiguities in the transmitters' associations appear due to symmetries in the transmitters' positions. In fact, the performance of those map matching methods in nearly symmetric scenarios is very poor.

In the present chapter, the Channel-SLAM map concept is augmented with information about the transmitters' visibility areas in order to improve the actual map matching methods. Firstly, the transmitter visibility concept and its storage during the algorithm operation are explained in Section 3.1. Secondly, a useful visibility metric for map comparison is created in Section 3.2. Finally, the visibility information is used to improve the classical map matching methods in Section 3.3.

3.1.- Storage: Map Estimation

The visibility regions of a transmitter are those areas of the scenario where the transmitter is in a LOS situation to the user. This visibility information can be included in the Channel-SLAM maps. Then, the additional information is expected to resolve some map ambiguities in the map matching process.

For breaking down the complexity of the map estimation, a grid map is used to discretize the two-dimensional space. Most common matrix cells are the triangular, quadrangular and hexagonal grids because of their perfect coupling without overlapping and without leaving empty space in between. In particular, we select the hexagon grid due to its greater resemblance to the circle, being able to better fit a real scenario. Nevertheless, a different grid could be chosen.

It is possible to better suit the environment by decreasing the hexagon size. However, a hexagon size reduction leads to a growth of the number of hexagons. In turn, this increases the computational and space complexity. Therefore, there is a trade-off between the accuracy in the visibility maps and the computational complexity.

Continuing with the visibility information storage, the fundamental idea is to store visibility information of every transmitter in each hexagon of the grid. This concept is depicted in Figure 3.1. The floor plan of an indoor scenario, where the black lines are walls, is outlined in Figure 3.1 (a) along with the visibility areas of a single transmitter Tx. A discretization of the scenario is depicted in Figure 3.1 (b): a hexagon map containing ideal visibility information of Tx is drawn over the floor plan. Finally, the ideal Channel-SLAM map related to the transmitter Tx is represented in Figure 3.1 (c): the map contains both the position and visibility areas of the transmitter.

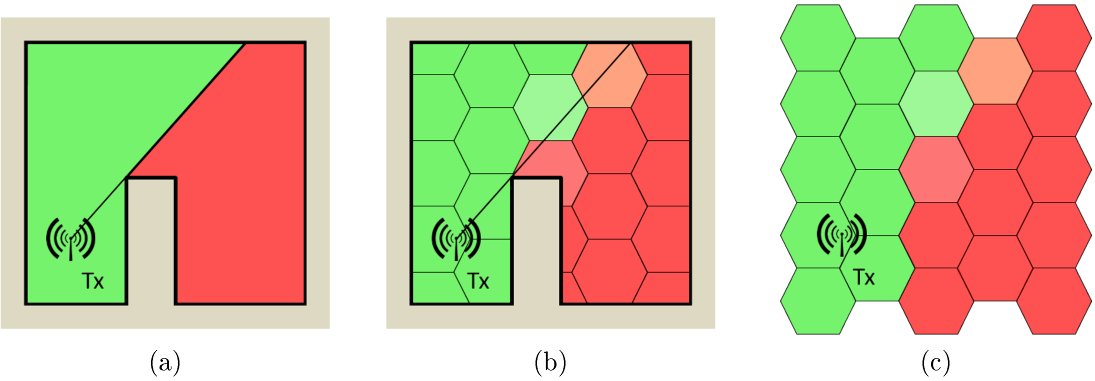


Figure 3.1.- Visibility information storage. The black lines represent walls, while the colors represent the observed transmitter visibility likelihood: from green/visible to red/non-visible.

It should be underlined that each hexagon stores the visibility information of all the transmitters, while in the depicted representation, for a better understanding, only the visibility regions of a single transmitter are drawn.

The hexagon map $\boldsymbol{M}$ is defined as the set

$$\boldsymbol{M} = \{\boldsymbol{M}^{<1>}, \boldsymbol{M}^{<2>}, ..., \boldsymbol{M}^{<h>}, ..., \boldsymbol{M}^{<N_{\text{hex}}>}\}, \tag{3.1}$$

where N_{hex} is the number of hexagons of the map and $\boldsymbol{M}^{<h>}$ represents the visibility information contained in the h^{th} hexagon. Each hexagon is defined in turn as

$$\boldsymbol{M}^{<h>} = \{M^{<h,1>}, M^{<h,2>}, ..., M^{<h,j>}, ..., M^{<h,N_{\text{TX}}>}\}, \tag{3.2}$$

where $M^{<h,j>}$ is the visibility probability of the j^{th} transmitter in the h^{th} hexagon.

Since $M^{<h,j>}$ is unknown, we denote $m^{<h,j>}$ the random variable whose conditional distribution at time instant k can be estimated through the history of the user state and the measurement vectors. The TOA and AOA measures themselves are not necessary, but the measurement vector is needed to determine which transmitters were seen at the different time instants.

With this, the Channel-SLAM mapping problem has changed from the estimation of the posterior PDF $\mathrm{p}\left(\boldsymbol{x}_k|\boldsymbol{z}_{1:k}\right)$ shown in Equation (2.16), to the estimation of

$$\mathrm{p}\left(\boldsymbol{x}_{0:k}, \boldsymbol{m}_k|\boldsymbol{z}_{1:k}\right) = \mathrm{p}\left(\boldsymbol{x}_{0:k}|\boldsymbol{z}_{1:k}\right)\mathrm{p}\left(\boldsymbol{m}_k|\boldsymbol{x}_{\mathrm{u},0:k}, \boldsymbol{z}_{1:k}\right), \tag{3.3}$$

where $\boldsymbol{m}$ represents the visibility map estimation and is defined as

$$\boldsymbol{m} = \{\boldsymbol{m}^{<1>}, \boldsymbol{m}^{<2>}, ..., \boldsymbol{m}^{<h>}, ..., \boldsymbol{m}^{<N_{\text{hex}}>}\}, \tag{3.4}$$

and $\boldsymbol{m}^{<h>}$ is the visibility estimation in the h^{th} hexagon

$$\boldsymbol{m}^{<h>} = \{m^{<h,1>}, m^{<h,2>}, ..., m^{<h,j>}, ..., m^{<h,N_{\text{TX}}>}\}. \tag{3.5}$$

The division of the space into independent hexagons makes it possible to decompose the visibility mapping problem into subproblems:

$$\mathrm{p}\left(\boldsymbol{m}_k|\boldsymbol{x}_{\mathrm{u},0:k},\boldsymbol{z}_{1:k}\right)=\prod_{h=1}^{N_{\mathrm{hex}}}\mathrm{p}\left(\boldsymbol{m}_k^{<h>}|\boldsymbol{x}_{\mathrm{u},0:k},\boldsymbol{z}_{1:k}\right). \tag{3.6}$$

Additionally, assuming independence among transmitters' visibility regions makes it possible, in turn, to decompose Equation (3.6) into

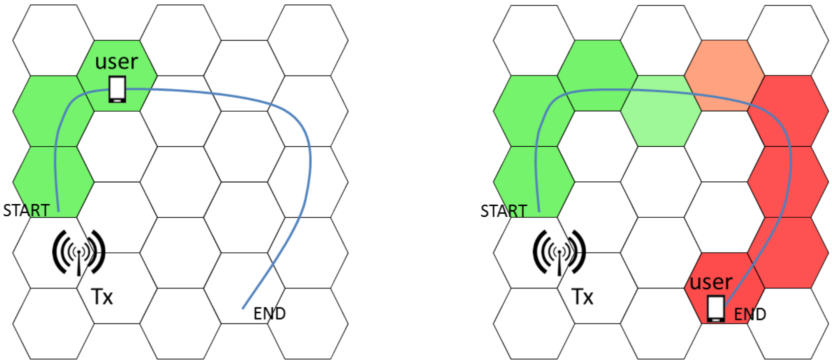
$$\mathrm{p}\left(\boldsymbol{m}_k|\boldsymbol{x}_{\mathrm{u},0:k},\boldsymbol{z}_{1:k}\right)=\prod_{h=1}^{N_{\mathrm{hex}}}\prod_{j=1}^{N_{\mathrm{TX}}}\mathrm{p}\left(m_k^{<h,j>}|\boldsymbol{x}_{\mathrm{u},0:k},\boldsymbol{z}_{1:k}^{<j>}\right). \tag{3.7}$$

Finally, the full posterior up to time instant k of the whole SLAM problem can be written as

$$\mathrm{p}\left(\boldsymbol{x}_{0:k},\boldsymbol{m}_k|\boldsymbol{z}_{1:k}\right)=\mathrm{p}\left(\boldsymbol{x}_{0:k}|\boldsymbol{z}_{1:k}\right)\prod_{h=1}^{N_{\mathrm{hex}}}\prod_{j=1}^{N_{\mathrm{TX}}}\mathrm{p}\left(m_k^{<h,j>}|\boldsymbol{x}_{\mathrm{u},0:k},\boldsymbol{z}_{1:k}^{<j>}\right), \tag{3.8}$$

where the obtainment of $\mathrm{p}\left(\boldsymbol{x}_{0:k}|\boldsymbol{z}_{1:k}\right)$ has been addressed in Subsection 2.2.2.

The Channel-SLAM algorithm assumes that no prior knowledge about the environment is known in advance. Thus, each user hexagon map is initially empty, i.e. no visibility information is available. During its path, the user estimates the probabilistic visibility map by adding visibility information to the hexagons traveled through. This is illustrated in Figure 3.2 where, again, only the visibility information of a single transmitter is represented.



(a) The user has visited three hexagons (b) The user has finished its track

Figure 3.2.- The user estimates the map while traveling through the scenario. The blue line represents the user track, while the hexagon colors represent the observed transmitter visibility likelihood: from green/visible to red/non-visible.

As stated in Chapter 2, every user particle independently estimates its own map. This means that each user particle autonomously estimates the location of the transmitters, and also creates its own visibility hexagon map. Finally, all user particle maps are combined in the map merging process.

How the visibility information is stored in each user particle map is explained in the following. Furthermore, how all user particle hexagon maps are combined is also described in detail.

Every hexagon of each user particle map contains two counters per transmitter. These counters $V_Y^{<i,h,j>}$ and $V_N^{<i,h,j>}$ store the number of times that the j^{th} transmitter has been seen and not seen, respectively, while the i^{th} user particle was located in the h^{th} hexagon of the map.

Nevertheless, these counters are not increased in every Channel-SLAM time step. They are only increased whenever a particle enters a new hexagon or, while being in the same hexagon, the visibility transmitter state changes from visible to non-visible or vice versa. In this way, if a user remains static the counters are not incremented to avoid increasing the map reliability. This also isolates the map estimation from the Channel-SLAM update rate.

Finally, in the map merging process all user particle maps are combined. In the merged map, the j^{th} transmitter visibility information contained in the h^{th} hexagon is given by the weighted visibility counters

$$W_Y^{<h,j>} = \sum_{i=1}^{N_{\mathrm{p,u}}} w_k^{<i>} V_Y^{<i,h,j>} \tag{3.9}$$

and

$$W_N^{<h,j>} = \sum_{i=1}^{N_{\mathrm{p,u}}} w_k^{<i>} V_N^{<i,h,j>}, \tag{3.10}$$

where $w_k^{<i>}$ is the i^{th} user particle weight at the time merging instant k. Thus, the particle reliability is taken into account.

3.2.- Usage: Map Visibility Metric

The Channel-SLAM feature map has been augmented. Now, a map contains, along with the position of the transmitters, a hexagon grid with visibility information. Therefore, the Channel-SLAM map has become a hybrid map.

It becomes necessary to develop a way to compare the visibility information of two grid maps in order to improve the map matching process. In this sense, a map visibility metric is developed.

From here on, the maps that are combined are named user and prior map. The user map is the augmented map that is estimated by the user, while the prior map has been obtained from another user and it is in a different coordinate system. The prior map obtained through crowd-sourcing could have been previously combined, containing information from preceding users and being more reliable.

Assuming that the transformation parameters among the two maps, i.e. rotation and translation, have already been calculated, the map visibility metric, also called visibility mismatch ratio, can be obtained following three simple steps.

Firstly, the number of common or overlapped hexagons $N_{\text{hex},common}$ is obtained. Secondly, the common hexagons in which the visibility information strongly differs are marked: $N_{\text{hex},marked}$. Lastly, the visibility mismatch ratio is obtained by means of

$$V_{ratio} = \frac{N_{\text{hex},marked}}{N_{\text{hex},common}}. \tag{3.11}$$

The process is depicted in Figure 3.3 for a better understanding.

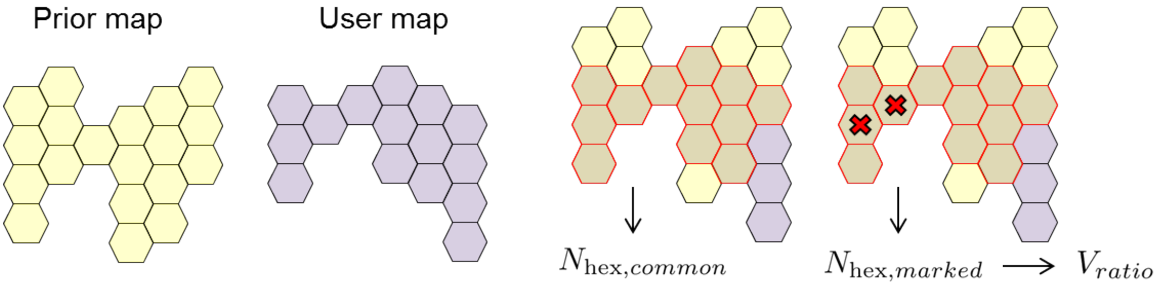


Figure 3.3.- Map visibility metric obtainment. The transformation parameters between maps are assumed as known, or already calculated. Overlapping and marked hexagons are highlighted with red edges and red crosses, respectively.

Thereby it becomes possible to determine the reliability of the map matching parameters among two maps through the visibility metric, in terms of transmitters' visibility. However, a way to determine when the information of two hexagons strongly differs needs to be found to perform the second step of the visibility metric calculation.

As described in Section 3.1, each hexagon stores two weighted visibility counters per transmitter, given by Equations (3.9) and (3.10). These weighted counters are used to create the hexagon visibility metric

$$H_{ratio}^{<h,j>} = \frac{W_Y^{<h,j>} + \frac{\alpha^{<h,j>}}{2}}{W_Y^{<h,j>} + W_N^{<h,j>} + \alpha^{<h,j>}}, \tag{3.12}$$

where h and j are the hexagon and transmitter indices, respectively, and

$$\alpha^{<h,j>} = \frac{1}{W_Y^{<h,j>} + W_N^{<h,j>}}. \tag{3.13}$$

The created ratio $H_{ratio}^{<h,j>}$ is used to estimate the visibility probability of the j^{th} transmitter in the h^{th} hexagon, $M^{<h,j>}$.

The alpha terms are included in Equation (3.12) to take into account the map reliability. The values of the hexagon weighted counters for a transmitter are high when a user has crossed the hexagon multiple times, or when the map has been previously combined and its information comes from different users. In this situation, the estimate for the transmitter visibility is reliable and the alpha terms become negligible. On the contrary, the alpha therms make the visibility estimate more robust if the hexagon has been visited only few times.

The developed hexagon ratio provides a fast way to perform the second step of the map visibility metric. Essentially, when comparing the h^{th} common hexagon from user and prior maps, the h^{th} hexagon ratios regarding all associated transmitters are compared. The common hexagon is not marked only if

$$\left| H_{ratio,\,\mathrm{u}}^{<h,j>} - H_{ratio,\,\mathrm{p}}^{<h,j>} \right| < H_{threshold}, \quad \forall\; j \in \{1, 2, ..., N_{\mathrm{TX}}\}, \tag{3.14}$$

where $|\cdot|$ is the absolute value function, and $H_{ratio,\,\mathrm{u}}^{<h,j>}$ and $H_{ratio,\,\mathrm{p}}^{<h,j>}$ are the user and prior maps' h^{th} hexagon visibility ratios, respectively, for the j^{th} transmitter. Otherwise, the common hexagon visibility information from both maps is regarded as different, and the h^{th} common hexagon of the combined map is marked.

3.3.- Application: Map Matching

The visibility mismatch ratio V_{ratio} allows to quantify how similar two maps are in terms of visibility information. There are two alternatives for the inclusion of this metric in the map matching process.

- Develop new map matching methods only based on visibility

  This option relies on a correlation to find the best transformation parameters between two maps, in terms of transmitters' visibility. The key idea is to find the match with the lowest visibility mismatch ratio and the highest number of common or overlapped hexagons.

- Include a check stage based on visibility in classical map matching methods

  This alternative consists in using the visibility information to validate the transformation parameters obtained by an already developed map matching method.

The obtainment of the visibility metric requires to go through user and prior hexagon maps to compare their information. Typically, the number of hexagons of a map is much higher than its number of transmitters, which makes a map matching approach based on the visibility information much more complex than a classic method based on the relative position of the transmitters. This is the main disadvantage of the first alternative presented. Another drawback comes with the requirement of an overlapping between user and prior hexagon maps to be performed.

Due to these disadvantages, the first alternative is discarded. A map matching method mainly based on visibility information has been developed and implemented in software, but it is not included in the present Master's thesis for a lack of interest.

For its part, the second alternative only calculates the visibility mismatch ratio once the transformation parameters have been obtained, in order to validate them with the visibility information of the maps. Thus, the computational complexity increase is much lower than in the first option. In addition, if the visibility metric cannot be obtained due to user and prior maps do not overlap, it is still possible to trust in the match estimated by the original map matching method.

Henceforth this section is focused on the second proposed alternative. Two map matching methods based on the relative position of the transmitters are described in Subsection 3.3.1, and the ambiguity problem of these methods is also explained with an example. The inclusion of the check stage based on visibility in already developed map matching methods is detailed in Subsection 3.3.2.

3.3.1.- Classical Map Matching Methods

As described in Section 1.1, classical map matching methods are only based on the transmitters' relative locations. For these methods, a map is only formed by the positions of well-estimated transmitters: transmitters with high variance in their location estimates are not used in the map matching process.

3.3.1.1.- Standard method

The first map matching method is extracted from [7], and hereafter is going to be referred as standard method. This method relies on comparing the distances between the transmitters of both maps to establish initial correspondences and perform the map matching. Its operation is briefly summarized in the following steps, and outlined in Figure 3.4, for a better understanding.

1. The number of transmitters NT used for obtaining the transformation parameters among the two maps is selected.

2. The Euclidean distances between the transmitters of each map are calculated. Going through all possible associations among NT transmitters from both maps, the best set of NT correspondences is obtained based on the distances between transmitters, following a least squares criterion.

3. Translation and rotation are estimated to match the NT associated transmitters from prior and user maps. The prior map is transformed.

4. After the prior map transformation, the positions of unassociated transmitters from prior and user maps are compared. If two unassociated transmitters are close enough, a new correspondence or association has been found.

5. Prior and user maps are combined. In this process all associated transmitters are merged. Unassociated transmitters are simply added to the combined map.

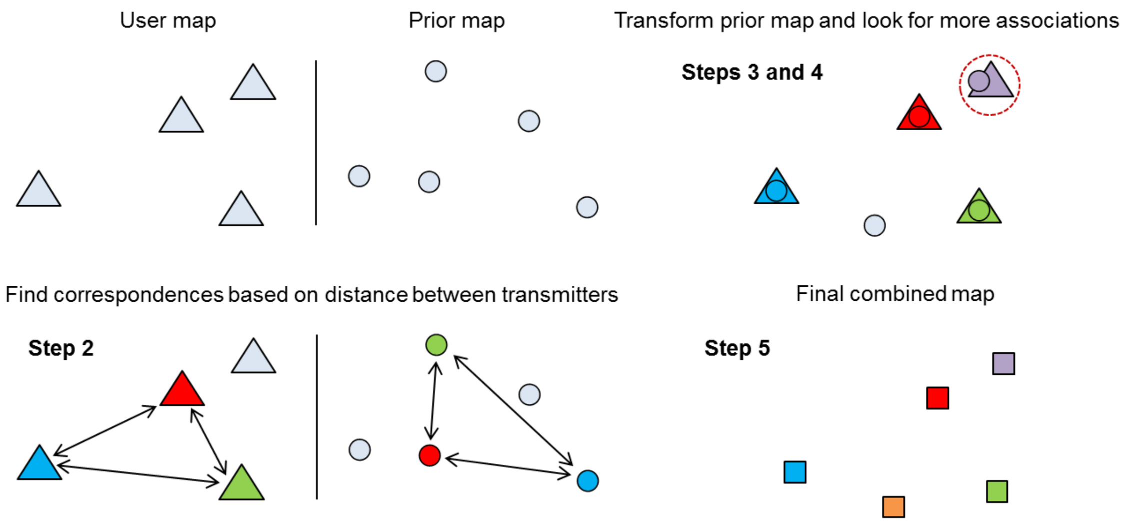

User map
Prior map
Transform prior map and look for more associations
Steps 3 and 4
Find correspondences based on distance between transmitters
Step 2
Final combined map
Step 5

Figure 3.4.- Standard method operation example, $NT = 3$. Transmitters are represented by triangles, circles and squares in the user, prior and combined maps, respectively. The transmitters' correspondences found by the method are represented by colors.

Since the transmitters' positions are represented by particle clouds, the mean positions are used for the distance calculation between transmitters in step 2. Furthermore, in the final map combination from step 5, the transmitters' particle clouds from associated transmitters are merged. Then, the grid resampling algorithm described in Subsection 2.2.3 is applied to reduce the number of particles.

3.3.1.2.- Rotation method

This map matching method has been specifically developed by the author of the present Master's thesis, and it is named rotation method because of its operation. It is based on finding the highest number of correspondences between transmitters from user and prior maps. The steps of this method are described as follows:

1. Randomly, one transmitter from the user and from the prior map are assumed to correspond: a transmitter association is established. Through this association, the translation between maps is obtained.

2. The prior map is translated and, using a certain rotation step, it is rotated along the 360 degrees. In each rotation step, the number of associated transmitters, i.e. transmitters from the prior map whose distance to a transmitter from the user map is lower than a certain distance threshold, is stored along with the translation and rotation applied.

3. Steps 1 and 2 are repeated along all possible transmitters' combinations, or until the number of associated transmitters is high enough. The transformation parameters that provide the highest number of associations are taken as a solution.

4. Prior and user maps are combined: associated transmitters are merged, while unassociated transmitters are plainly added to the combined map.

As in the standard method, the mean positions of the transmitters' particle clouds are used when finding correspondences in step 2. Again, the transmitters' particle clouds from associated transmitters are merged in step 4, and the grid resampling algorithm is applied to reduce the number of particles.

The rotation method operation is outlined in Figure 3.5, using the same user and prior maps from the standard method example. The final combination step is omitted.

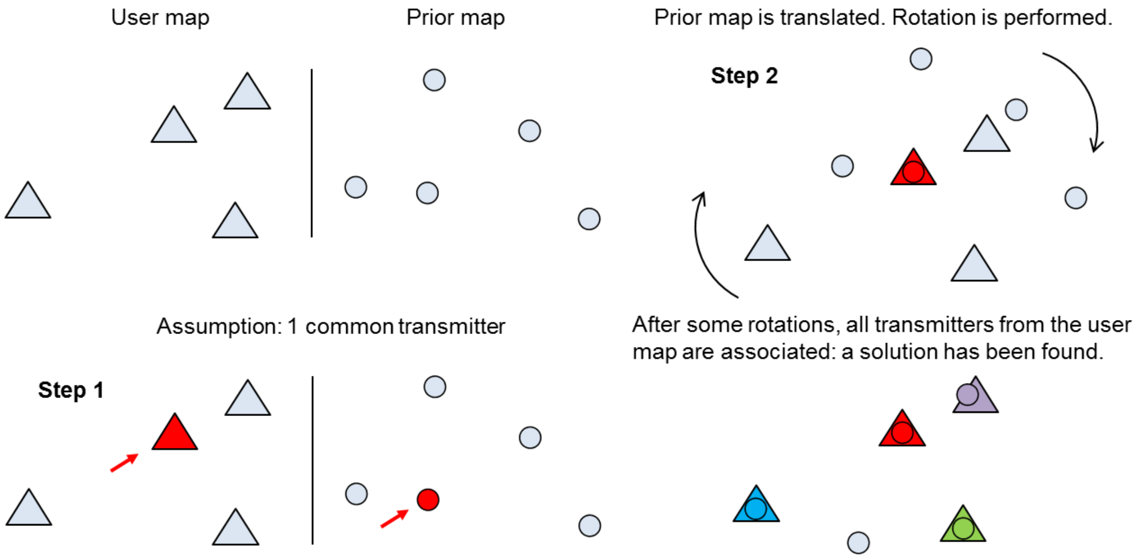

User map
Prior map
Prior map is translated. Rotation is performed.
Step 2
Assumption: 1 common transmitter
Step 1
After some rotations, all transmitters from the user map are associated: a solution has been found.

Figure 3.5.- Rotation method operation example. Transmitters are represented by triangles and circles in the user and prior maps, respectively. The transmitters' correspondences found by the method are represented by colors. The map combination is omitted.

3.3.1.3.- Classical methods problem

Map matching methods based on the relative position of the transmitters may fail when symmetries in the transmitters' locations exist. This problem can appear in common indoor scenarios, depending on the transmitters used for finding the transformation parameters, and it aggravates in nearly symmetrical scenarios, where almost every possible combination of transmitters presents ambiguities.

This ambiguity problem is shown through a simple example in Figure 3.6. The user map is formed by three transmitters with the same distances between them, while the prior map includes an additional transmitter, whose position could be useful to improve the user map. The problem resides in the lack of information to perform the map matching: it is not possible to determine which of the three possible map combinations is correct.

In this example, a wrong combination would introduce false information in the user map, which could lead into an erroneous performance of the whole algorithm. In addition, once the map combination is performed, the resulting map may be shared with other users, and the error could be propagated.

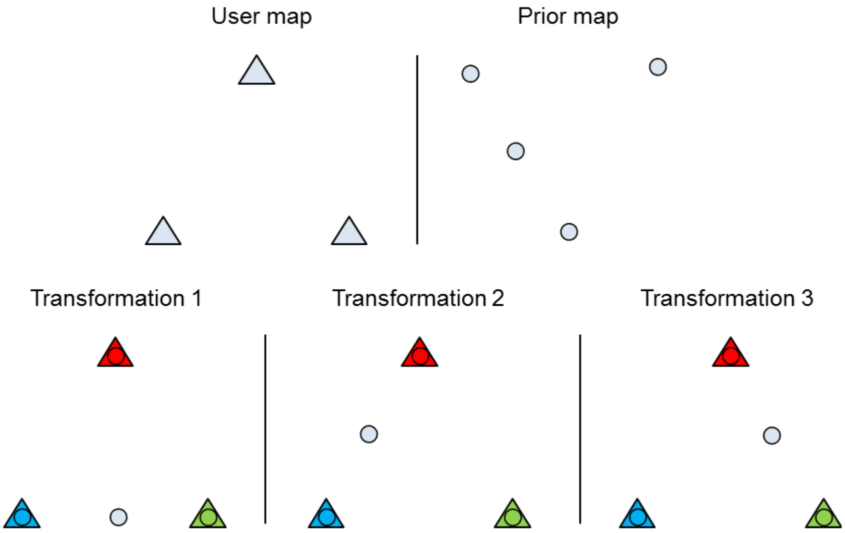


Figure 3.6.- Problem of the classical map matching methods: ambiguity in the transmitters' correspondences. Transmitters are represented by triangles and circles in the user and prior maps, respectively. Transmitters' correspondences are represented by colors.

The main motivation to introduce additional information of transmitters in the Channel-SLAM maps, as the visibility information, is to tackle the described ambiguity problem.

3.3.2.- Check Stage based on Visibility

For handling the ambiguity problem, the map visibility metric from Section 3.2 is incorporated into the final part of a classical map matching method. The operation of the resulting generic method is described as follows:

1. A classical map matching method is used to obtain an ordered list formed by estimated transformation parameters. The list is ordered following the classical method criterion.

2. Using the first matching parameters of the list, the whole prior map, i.e. transmitters' positions and hexagon map, is translated and rotated.

3. The visibility mismatch ratio V_{ratio} between the user and the transformed prior maps is calculated and compared with a visibility threshold $V_{threshold}$.

   If the ratio is higher than the threshold, the method goes back to step 2, using in this case the next matching parameters of the list. Otherwise, a solution has been found.

4. Prior and user maps are combined: associated transmitters are merged, unassociated transmitters are simply incorporated into the combined map, and prior and user hexagon maps are added.

The described generic method is compatible with every classical map matching method based on transmitters' locations. The usage of different classical methods in step 1 leads to distinct map matching solutions because they follow a different criterion.

In order to facilitate the comparison of the hexagon maps, the same fixed hexagon grid is used to store the visibility information. In the hexagon map transformations, only the centers of the hexagons are rotated and translated; the grid remains static. Then, the information contained in the original hexagons is incorporated into the hexagons of the fixed grid where the hexagon centers fall after being transformed. This is, a hexagon map transformation involves the rotation and translation of the information contained in the map, not the rotation of the hexagons themselves.

The visibility threshold $V_{threshold}$ used in step 3 represents the allowed number of overlapped hexagons in which the visibility information from the user and prior map mismatches, referred to the total number of overlapped hexagons. It becomes possible to control how restrictive the method is through its value.

The map combination from step 4 is actually carried out during the visibility mismatch ratio calculation from step 3: both operations require to go through the user and prior hexagon maps, being possible to be performed at the same time. However, they have been listed separately because they represent different tasks.

Furthermore, the hexagon map combination performed in step 4 involves a visibility information combination. Non-common hexagons from user and prior maps are plainly incorporated into the combined map, and their visibility information is not modified. Nevertheless, the h^{th} common hexagon weighted counters of the combined map regarding the j^{th} transmitter are given by the sum of the counters of the individual maps as follows:

$$W_{Y,\mathrm{c}}^{<h,j>} = W_{Y,\mathrm{u}}^{<h,j>} + W_{Y,\mathrm{p}}^{<h,j>}, \tag{3.15}$$

$$W_{N,\mathrm{c}}^{<h,j>} = W_{N,\mathrm{u}}^{<h,j>} + W_{N,\mathrm{p}}^{<h,j>}, \tag{3.16}$$

where the subscripts c, u and p are referred to the combined, user and prior maps, respectively. These combinations represent the hexagon map reliability increase.

Going back to the example depicted in Figure 3.6, the generic map matching method described in this subsection would be able to obtain the correct combined map if the visibility areas of at least one of the used transmitters differ from the rest.

In other map combinations where the described ambiguity problem is not present and the solution provided by the map matching method is not modified by the check stage, the visibility metric can be useful for other purposes. For example, its value can be interpreted as a reliability metric of the combined map, and used for comparing the performance of different classical map matching methods.

4.- Evaluations and Results

The practical developed work is described in this chapter. Simulations are used to evaluate the performance of the map matching methods with the inclusion of the check stage based on visibility information from Subsection 3.3.2. The MATLAB software is used to carry out such simulations.

In order to perform the map matching, maps need to be generated. This process is described in Section 4.1. In Section 4.2, the map matching performance of the classical methods with and without the check stage are compared. The influence of the hexagon size in the map matching performance is also evaluated. Finally, the complexity of the developed check stage is studied in Section 4.3.

4.1.- Estimation of Maps

A simplified version of Channel-SLAM has been implemented to obtain several maps. As described in Chapter 2, the Channel-SLAM approach is composed of two stages: the parameter and the position estimation. The second stage performs SLAM using the parameters of the signal components estimated in the first stage.

The conducted implementation of Channel-SLAM corresponds to the second stage of the method: position estimation. As inputs, AOA and TOA estimates are created after adding AWGN to their true value. Furthermore, measurements coming from an IMU are emulated with the addition of AWGN to the true user speed and heading.

The visibility map estimation described in Section 3.1 is also incorporated into the Channel-SLAM implemented version. Therefore, the hexagon maps are created and augmented during the algorithm operation, and the transmitters' visibility information can be used in the map matching process for solving ambiguities among transmitters' associations. As already explained, such ambiguities are caused by symmetries in the transmitters' positions, which in turn can be caused by symmetries in the scenario.

The final objective of the performance analysis is to observe the effect of the visibility information inclusion in the map matching process. Thus, a nearly symmetric indoor scenario is created to perform the simulations. Figure 4.1 shows a top view of the challenging scenario, where black lines represent reflecting walls, and the red triangle labeled Tx0 is a physical transmitter. Single reflections of the signal broadcasted by Tx0 create the virtual transmitters represented by the pink triangles. Virtual transmitters are labeled with the number of the wall that interacts with Tx0 to create them. Users can only move within the depicted room.

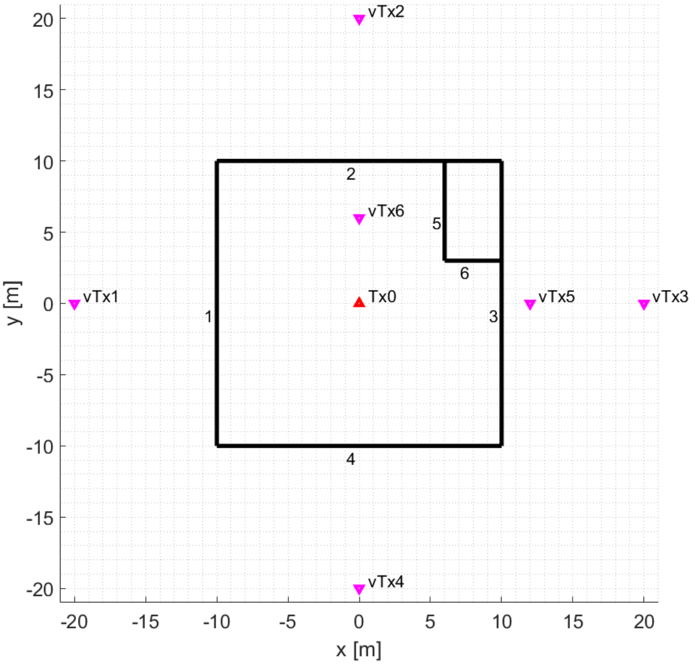
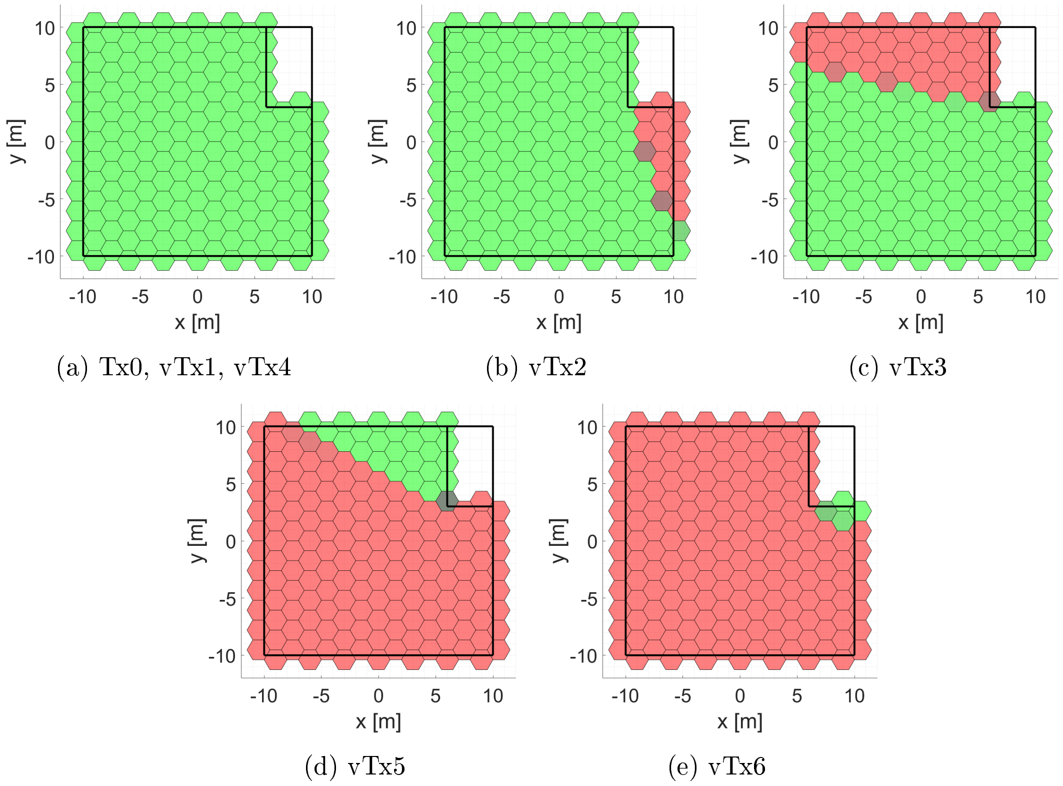


Figure 4.1.- Nearly symmetric indoor scenario. Black lines represent reflecting walls, while physical and virtual transmitters are outlined by red and pink triangles, respectively.

The physical transmitter Tx0 is located in the center of the room, at the same distance from walls 1 to 4, creating symmetries among virtual transmitters vTx1 to vTx4. Thus, if the transformation parameters are obtained only using the positions of the physical transmitter Tx0 and the virtual transmitters vTx1 to vTx4, or a subset of them, ambiguities in the transmitters' correspondences appear and the map matching may fail. The usage of vTx5 and/or vTx6 can solve the ambiguity, but they are only seen in a small region of the map. It is thus likely that the position uncertainty for these two transmitters is high, and they cannot be used in the map matching process, as explained in Subsection 3.3.1. Therefore, a poor performance of the classical map matching methods is expected in this scenario.

To provide a complete description of the simulation scenario, the true discretized visibility areas of the transmitters are outlined in Figure 4.2, using a hexagon side length of 1 meter. Transmitters Tx0, vTx1 and vTx4 can be seen in the whole room, and they cannot be distinguished by means of the visibility information. Nevertheless, the obstacle from the top right makes the visibility areas of transmitters vTx2 and vTx3 to differ from the rest and from each other. Depending on the user trajectory, the visibility information can make the difference in the map matching process and solve the association ambiguities.

As outlined in Figure 4.2 (e), the visibility area of vTx6 is very small. Even if the user travel through that area, the transmitter particles may not converge to the transmitter position as described in Subsection 2.2.3. High variance position estimates are not used in the map matching process; consequently, vTx6 is not expected to be used in any transformation parameters obtainment.



(a) Tx0, vTx1, vTx4

(b) vTx2

(c) vTx3

(d) vTx5

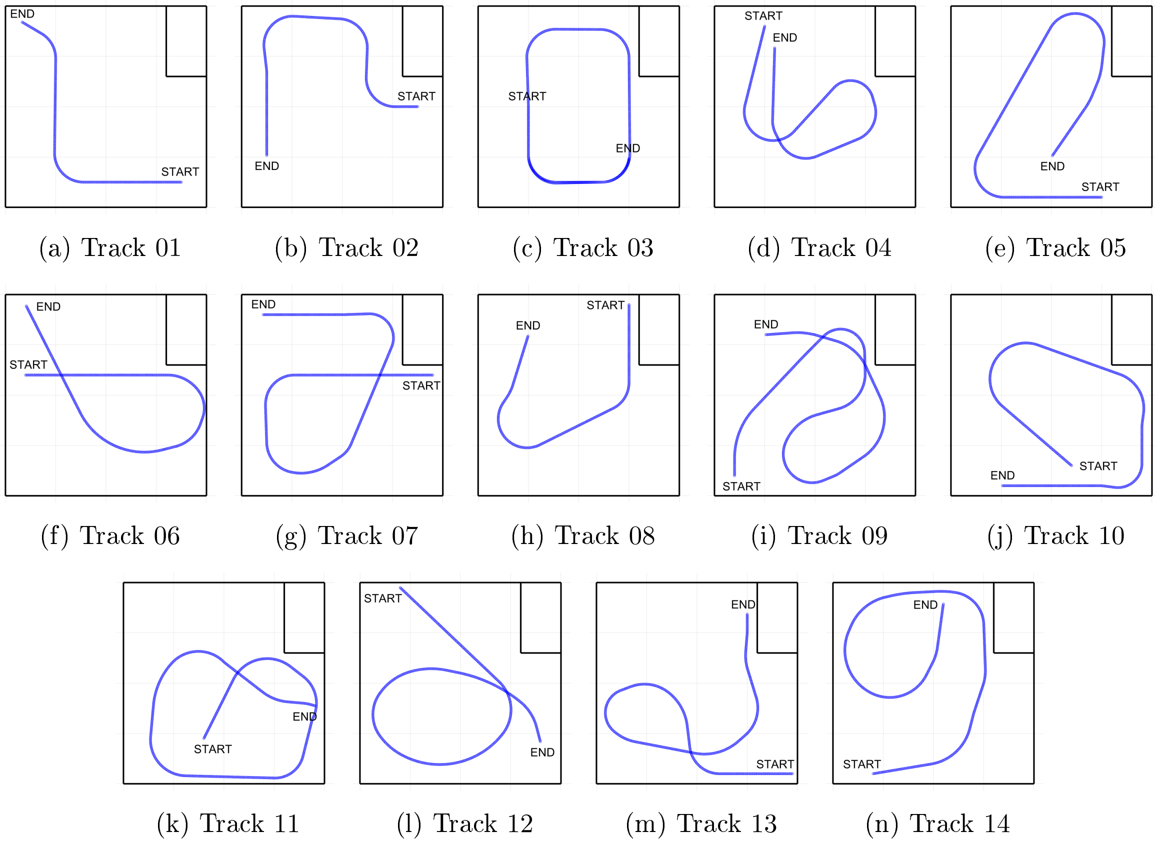
(e) vTx6

Figure 4.2.- The actual discretized transmitters' visibility areas are drawn over the simulation scenario floor plan. The hexagon side length used is 1 meter. Hexagon colors represent the transmitter visibility likelihood: from green/visible to red/non-visible.

The discretization of the space allows to reduce the complexity of the visibility mapping problem, but the borders between visible and non-visible regions cannot be perfectly suited. As explained in Section 3.1, there is a trade-off between the accuracy in the visibility maps and the computational complexity: the lower the hexagon size, the better the accuracy in the visibility information and the higher the map managing computational complexity.

An objective of the present evaluation chapter is to show such trade-off existence through simulation results. For that purpose, the implemented Channel-SLAM algorithm simultaneously estimates the transmitters' visibility areas through three hexagon maps with different hexagon sizes. At the end of a Channel-SLAM simulation, three maps with exactly the same transmitters' position estimates but different visibility information are obtained. Then, fair comparisons between the map matching results using different hexagon sizes can be made. In particular, the hexagon side lengths used in simulations are 2, 1 and 0.5 meters.

In order to obtain several maps to perform multiple map matching evaluations, 14 user trajectories with different length, area covered and geometry are created. Such tracks are drawn in blue over the indoor scenario floor plan in Figure 4.3. Altogether, 15 maps are generated per hexagon size: 14 are created through the Channel-SLAM simulations over these trajectories, and the remaining one is the true map.



(a) Track 01 (b) Track 02 (c) Track 03 (d) Track 04 (e) Track 05

(f) Track 06 (g) Track 07 (h) Track 08 (i) Track 09 (j) Track 10

(k) Track 11 (l) Track 12 (m) Track 13 (n) Track 14

Figure 4.3.- The user tracks are drawn in blue over the indoor scenario floor plan. Each track starting and final points are labeled START and END, respectively.

In every simulation each user moves with a constant speed of 1 m/s. TOA and AOA estimates are obtained every 100 ms, along with speed and heading estimates. No prior knowledge of the environment is known by the user, i.e. walls and transmitters' positions are assumed as unknown.

An example of a Channel-SLAM simulation is shown in Figure 4.4. The track 10 from Figure 4.3 (j) is used. The represented illustration corresponds to the last time step of the simulation: the true position of the user at the end of the simulation is sketched as a red cross, while its position estimation is represented by the yellow particle cloud. The true and estimated user trajectories are depicted in blue and dark red, respectively. The estimated map, i.e. transmitters' particle clouds and visibility hexagon map, is also drawn over the floor plan. The side length of the represented hexagons is 1 meter, and they are not filled for a better visualization; however, visibility information of all the transmitters is contained in them. Different colors have been selected to distinguish the particle clouds that estimate the position of each transmitter.

As it can be seen, the virtual transmitters vTx5 and vTx6 are not represented nor estimated since they are never seen by the user. This means that classical map matching methods may fail when using the obtained map in the process, whether as user or prior map, due to symmetries in the estimated positions of the transmitters vTx1 to vTx4.

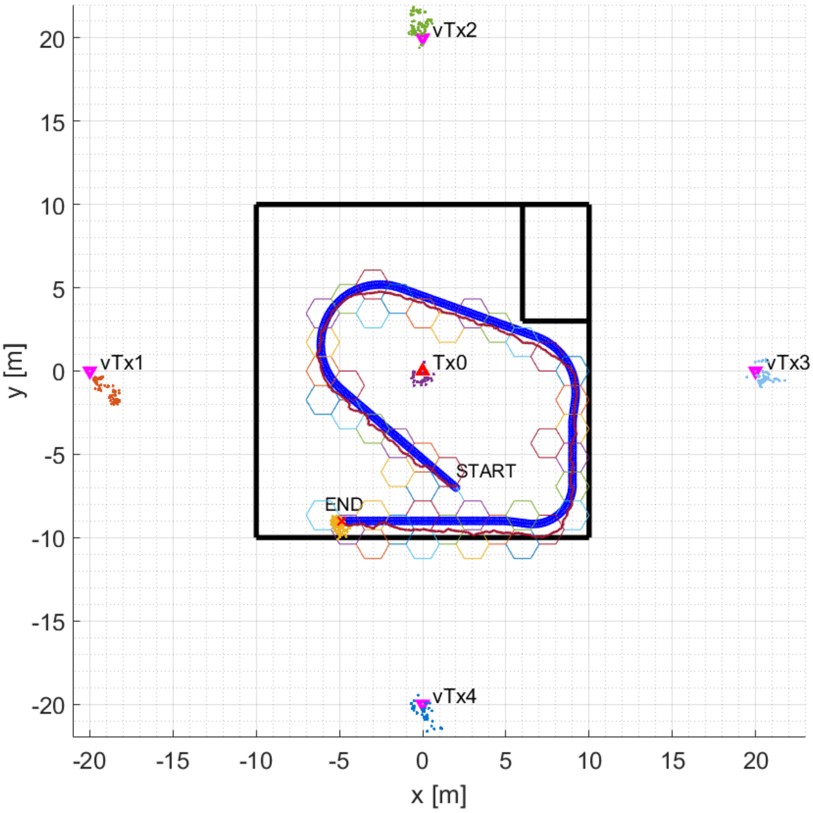
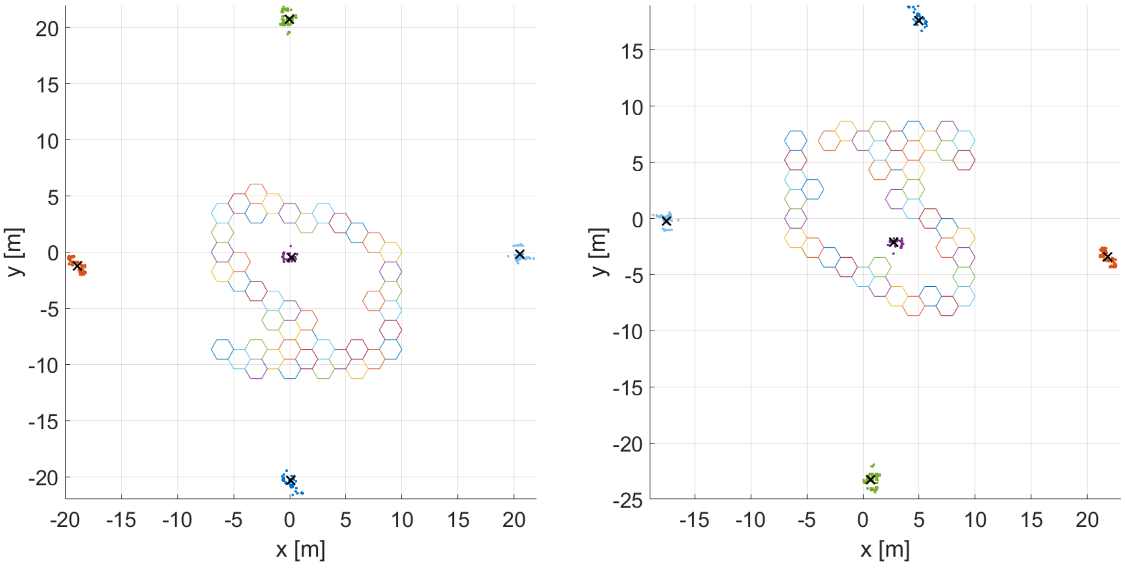


Figure 4.4.- Last step of a Channel-SLAM simulation over the track 10. Black lines are reflecting walls, and the physical and virtual transmitters are represented by the red and pink triangles, respectively. The true user position at the last time step is denoted by a red cross, while the estimated user position is depicted by the yellow particle cloud. The true and estimated user tracks are drawn in blue and dark red, respectively. The estimated map, formed by the transmitters' particle clouds and the visibility hexagon map, is also drawn.

In the previous simulation, the user initial position and heading were assumed as known to properly draw the estimated map, user position and trajectory over the floor plan. However, at the end of the simulation a randomly rotated and translated version of the map is stored, as if the initial position and heading were unknown. The randomly transformed maps are the ones used in the map matching evaluations in future sections, as if every user were in its own local coordinate system.

Figure 4.5 shows the map obtained in the previous simulation before and after being randomly transformed, i.e. assuming that the user initial position and heading were available and unavailable, respectively. A different color has been used to draw the position estimation of each transmitter, while the transmitters' mean position estimates are depicted by black crosses.

Again, each hexagon of the drawn visibility maps has a side length of 1 meter. However, the maps with hexagon side lengths of 2 and 0.5 meters obtained in the Channel-SLAM simulation are also transformed using exactly the same rotation and translation parameters. In this way, the map matching performance using a different hexagon size can be fairly compared in future sections.



(a) Map

(b) Randomly transformed map

Figure 4.5.- Channel-SLAM maps obtained from the simulation depicted in Figure 4.4. In (a), the initial position and heading of the user are known. In (b), the original map is randomly transformed to emulate unknown initial user position and heading.

The map depicted in Figure 4.5 (b) is the information that a specific user, who uses a certain local coordinate system, has about the environment. In the original conception of Channel-SLAM, a map was only composed of the transmitters' positions. Now, visibility information of every transmitter is stored in each hexagon of the visibility grid map.

4.2.- Map Matching Performance

The main objective of this section is to compare the performance of the classical map matching methods with and without the inclusion of the check stage based on visibility from Subsection 3.3.2. This evaluation is carried out in Subsection 4.2.1, while the influence of the hexagon size in the map matching performance is analyzed in Subsection 4.2.2.

After conducting the Channel-SLAM simulations over the 14 user tracks, and taking into account the true map of the scenario, 15 maps are available per hexagon size to evaluate the map matching methods performance. Using these maps as user and prior maps in the map matching process, 105 map combinations can be carried out per hexagon size and for each map matching method.

The number of successful map combinations out of the 105 total combinations is used to evaluate the performance of the different map matching methods with and without the check stage, and with different configurations. This ratio is called success rate hereafter. The success criterion used to obtain the verdict on each map combination is specified as follows.

- If the transmitters' associations among user and prior maps provided by the map matching method are wrong, the map combination is regarded as unsuccessful.
- In case a transmitter correspondence is not detected, leading to a transmitter duplicity in the combined map, the map combination is regarded as unsuccessful.

  Nevertheless, if a certain transmitter correspondence is not detected but there is a big error in the user or prior maps regarding that transmitter, the error is not considered as a map matching error.

  The initial errors of the original 15 maps are detected after comparing them with a true map template. In particular, an error in a transmitter location is detected when it is further than 2 meters from its true position.
- The resulting combined map is also compared with the true map template. In case a transmitter of the combined map is further than 2 meters from its true position, the map combination is regarded as unsuccessful.

  Again, initial errors from the original maps that are being combined, i.e. user and prior maps, are not counted as map matching errors.
- In case the map matching cannot be performed by a certain classical method, the evaluation cannot be carried out neither with or without the check stage. Thus, the corresponding map combination is not taken into account in the success rate.

  This phenomenon can happen with certain methods' configurations. For example, using the standard method from Subsection 3.3.1.1 with NT transmitters, the map matching cannot be performed if user and prior maps do not contain reliable information of, at least, NT transmitters.
- In case a classical map matching method achieves a successful map combination, but the inclusion of the check stage makes the method to not rely on any solution, the verdict of the method with the check stage is unsuccessful.
- In case a classical map matching method achieves an unsuccessful map combination, and the inclusion of the check stage makes the method to not rely on any solution, the verdict of the method with the check stage is successful.

In summary, if a certain method finds a map combination, the correspondences among transmitters from user and prior maps are correct, and no errors in the combined map transmitters' positions caused by the map matching are found, the combination is regarded as successful. In case the check stage inclusion makes a certain method to stop considering its solution as reliable, the verdict of the method with the check stage is the opposed to the verdict of the method without the check stage.

4.2.1.- Check Stage Influence

Once the success criterion has been defined, the performance of the map matching methods described in Subsection 3.3.1 with and without the check stage based on visibility information is evaluated. In particular, the success rates of 4 methods with and without the check stage are compared in this subsection: rotation method, and standard method using $NT = 2$, 3 and 4 transmitters. For performing this comparison, the Channel-SLAM maps with a hexagon side length of 1 meter are used.

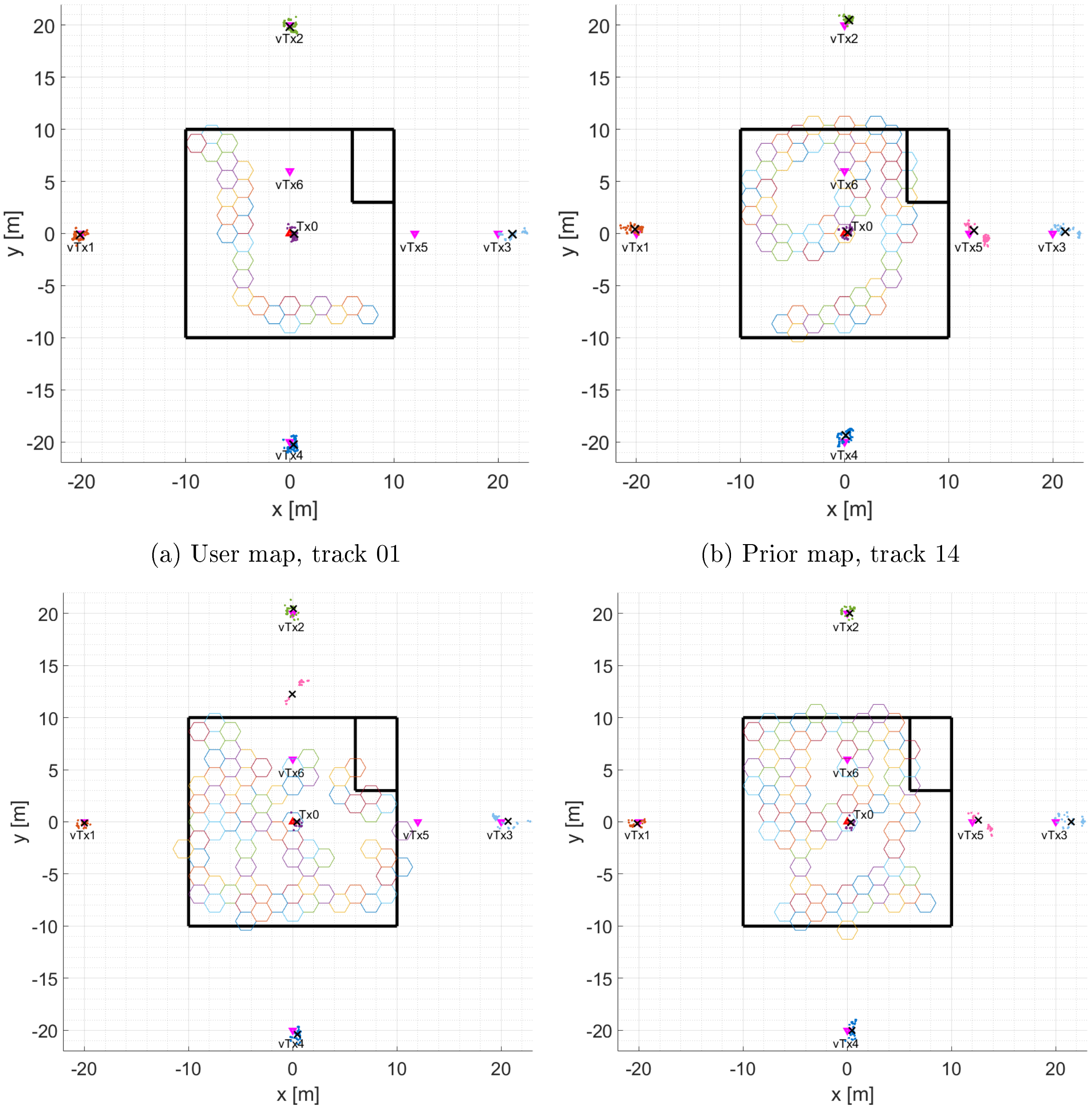
Before showing the global results, a single map matching simulation is depicted in Figure 4.6 as an example. The standard method with and without the check stage is used in this simulation, with $NT = 3$. The user and prior maps used in the map combination correspond to the tracks 01 and 14, respectively, from Figure 4.3. For a better visualization, they are represented without being rotated and translated in Figures 4.6 (a) and (b). Nevertheless, the randomly transformed versions of these maps are the ones used in the map matching.

The combined maps obtained by the standard method with and without the check stage are represented in Figures 4.6 (d) and (c), respectively. Even though the visibility hexagon maps are not used by the method without the check stage, they are drawn to highlight differences with the solution obtained by the method with the check stage.

As it can be seen in Figure 4.6 (a), the user map does not contain estimates for the transmitters vTx5 and vTx6. Thus, only the information of the transmitters Tx0 and vTx1 to vTx4 can be used. However, symmetries in the transmitters' estimated positions are present. These symmetries make the standard method without the check stage to fail because it only relies on the transmitters' locations: the combined map from Figure 4.6 (c) contains false information about the transmitter vTx5 location, which is estimated to be above the room instead of on its right hand side.

When using the standard method with the check stage, the transmitters' visibility information allows to distinguish the virtual transmitter vTx3 from the rest, since it cannot be seen in the upper part of the scenario. This, in turn, allows to solve the ambiguities in the transmitters' associations, and the transformation parameters can be estimated with a good accuracy. The combined map obtained by this method from Figure 4.6 (d) is a successfully augmented version of the user map from Figure 4.6 (a).

Following the success criterion, the map combination performed by the standard method without the check stage is unsuccessful because the associations among transmitters from user and prior map used for obtaining the transformation parameters are wrong. For its part, the map combination performed by the standard method with the check stage is successful: transmitters' associations are correct, there are no transmitters' duplicities, and the transmitters' estimated positions are close to their true position.



(a) User map, track 01

(b) Prior map, track 14

(c) Combined map, method without check stage

(d) Combined map, method with check stage

Figure 4.6.- Map combination example using the standard method with and without the check stage, with $NT = 3$. Different colors are used to distinguish the transmitters' positions estimates, and the visibility hexagon maps are also represented. The transmitters' mean position estimates are depicted by black crosses. In (c), the method fails due to the ambiguity introduced by the symmetric location of the transmitters vTx1 to vTx4, while in (d), the check stage usage leads to a successful match.

Finally, the performance results obtained after conducting the 105 map combinations per implemented method are shown in Table 4.1. The successful, unsuccessful and total number of combinations is specified in each case, along with the obtained success rate. The results of the map matching methods with the check stage based on visibility are emphasized in blue. As explained in the success criterion definition, some map combinations are not taken into account in the results of certain configurations of the methods, because the map matching could not be performed.

Map matching method	Successful	Unsuccessful	Total	Success rate
Rotation	19	86	105	18.10%
Rotation + Check stage	40	65	105	38.10%
Standard, $NT = 2$	31	74	105	29.52%
Standard, $NT = 2$ + Check stage	61	44	105	58.10%
Standard, $NT = 3$	27	75	102	26.47%
Standard, $NT = 3$ + Check stage	80	22	102	78.43%
Standard, $NT = 4$	32	62	94	34.04%
Standard, $NT = 4$ + Check stage	79	15	94	84.04%

Table 4.1.- Performance results of map matching methods obtained through simulations. The hexagon side length of the visibility maps is fixed to 1 meter. The results of the methods with the check stage based on visibility are highlighted in blue.

As expected, the performance of the map matching methods only based on the relative position of the transmitters is very poor in the simulated scenario due to the high amount of symmetries in the transmitters' positions. The standard method using 4 transmitters to obtain the transformation parameters only has a success rate of 34.04% percent, while the performance of the rest of the evaluated methods is even worse.

The inclusion of the check stage based on visibility considerably increases the success rate of every evaluated map matching method in this challenging scenario. The success rate gets doubled in the rotation and the standard method with $NT = 2$, while using $NT = 3$ it gets tripled. Finally, the success rate of the standard method using 4 transmitters increases by 50%, reaching the 84.04% of success. These results show that the inclusion of the check stage at the final part of a map matching method allows to avoid transmitters' correspondences ambiguities produced by symmetries in the transmitters' locations.

4.2.2.- Hexagon Size Influence

As already mentioned, the hexagon size controls a trade-off relation between the accuracy in the visibility information contained in a hexagon map, and its computational complexity. The goal of this subsection is to determine if the accuracy in the visibility information, i.e. the hexagon size of a visibility map, has an influence in the map matching performance on methods with the check stage.

Thus, the success rate of the standard method with the check stage is calculated using the maps with different hexagon sizes obtained in Section 4.1. These maps contain exactly the same transmitters' position estimates, but different visibility hexagon maps. In particular, 15 maps are available per hexagon size that allow to perform fair comparisons through the simulation results. The hexagon side lengths of these maps are 2, 1 and 0.5 meters.

The results obtained after performing all the map combinations with different configurations of the standard method are summarized in Table 4.2. The number of transmitters used by the standard method to find the transformation parameters varies from 2 to 4. The performance results using 1 meter as hexagon side length have been already shown in Table 4.1.

Configuration		Results			
NT	Hexagon side length	Successful	Failure	Total	Success rate
2	2 m	52	53	105	49.52%
3	2 m	72	30	102	70.59%
4	2 m	73	21	94	77.66%
2	1 m	61	44	105	58.10%
3	1 m	80	22	102	78.43%
4	1 m	79	15	94	84.04%
2	0.5 m	65	40	105	61.90%
3	0.5 m	82	20	102	80.39%
4	0.5 m	80	14	94	85.11%

Table 4.2.- Performance results of the standard map matching method with the check stage. The number of transmitters used in finding the transformation parameters is varied from 2 to 4, and three hexagon side lengths are considered: 2, 1 and 0.5 meters.

The success rates obtained show that the decrease of the hexagon size involves, thanks to the improvement in the visibility information accuracy, an improvement in the performance of the map matching methods with the check stage based on transmitters' visibility. This improvement is achieved in all evaluated configurations of the standard method: $NT = 2$, 3 and 4 transmitters.

When moving from using a hexagon side length of 2 meters to 1 meter, the performance improvement is significant: the success rate increases 6.38, 7.84 and 8.58 percentage points for $NT = 2$, 3 and 4, respectively. However, this improvement gets lowered to 1.07, 1.96 and 3.80 percentage points, respectively, when moving from 1 meter to 0.5 meters. This allows to suspect that at some point, with a specific hexagon size, the visibility information of the transmitters would be almost perfectly suited and better results could not be obtained. In fact, a hexagon size reduction may lead to a smaller overlapping between user and prior maps. If the hexagon size reduction makes the maps to not overlap, the comparison of the visibility information cannot be performed.

Finally, the comparison of the results from Tables 4.1 and 4.2 shows that even the usage of a hexagon side length of 2 meters in the transmitters' visibility estimation can considerably improve the performance of the classical map matching methods. A hexagon side length of 2 meters involves a hexagon area of 10.39 square meters.

Only around 40 hexagons of that size are needed to cover the whole scenario depicted in Figure 4.1. In this situation, the transmitters' visibility areas are not accurately suited, but the inclusion of the check stage still makes the difference on the map matching performance. For example, the success rate of the standard method with $NT = 4$ gets increased from 34.04% to 77.66%.

4.3.- Check Stage Complexity

In the previous section it has been shown that the inclusion of the check stage developed in the present Master's thesis can improve the performance of the classical map matching methods in challenging scenarios, where symmetries in the transmitters' positions arise. Nevertheless, the inclusion of the check stage at the end of a map matching method involves a computational complexity increase.

According to the operation of a generic map matching method with the check stage described in Subsection 3.3.2, the computational increase is clearly marked by two factors. The first one is the number of times that the check stage is performed, i.e. the number of times that a solution found by the method based on the position of the transmitters is rejected after checking the transmitters' visibility information. This is analyzed in Subsection 4.3.1. The second factor is the size of the user and prior maps that are being combined. Based on the size of the user and prior maps, the computational complexity of the check stage is theoretically studied and experimentally verified in Subsection 4.3.2.

4.3.1.- List Size Influence

In the operation of a map matching method with the check stage, the original method only based on the transmitters' positions is used to create a list with transformation parameters, ordered by its own reliability criterion. For example, the standard method criterion is based on finding the summed up minimum squared distance differences among NT associated transmitters from both maps. In such case, the list would be ordered by those distance differences. Then, using the parameters of the first entry in the list, the prior map is transformed to match the user map, and the visibility mismatch ratio is obtained. If the visibility information of the maps is similar, the solution is regarded as reliable and the maps are combined. In case the visibility information of the maps differs, the next entry in the list is evaluated. This process is repeated until a reliable solution is found, or until the last entry of the list is checked.

The number of times that the check stage is going to be performed cannot be determined. However, the size of the parameter list can be limited. In this subsection, the performance of the standard map matching method with the check stage is evaluated through simulations, varying the maximum size of the parameter list.

In order to analyze the simulation results, some aspects of the success criterion need to be emphasized. As explained in the criterion definition, it is not necessary for the map matching method with the check stage to obtain a combined map to be successful. If the original method reaches a wrong solution, and the check stage inclusion avoids that wrong solution, it is regarded as successful even though it does not provide a combined map, because it is still detecting and avoiding errors.

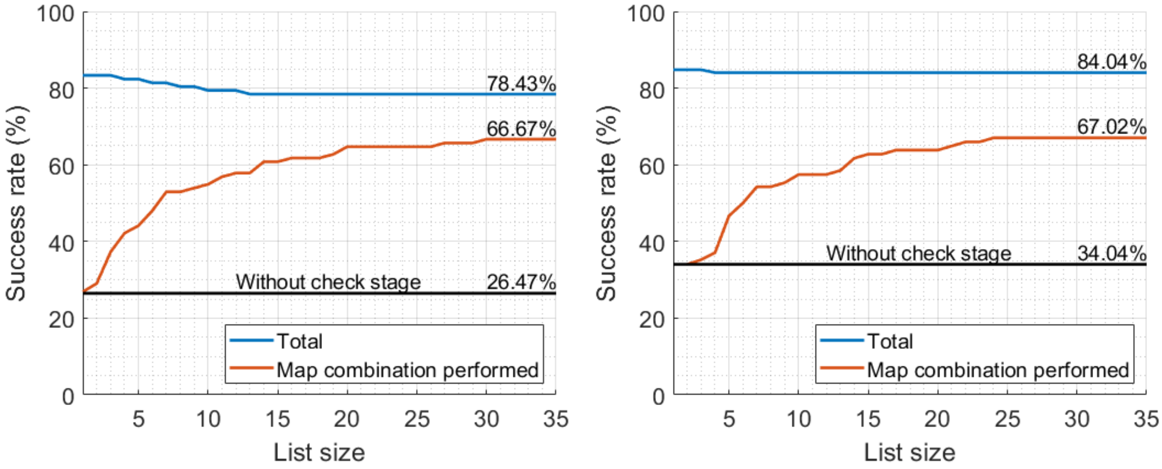
Such a case happens if the parameter list size is short. In most map combinations, the solution achieved by the method with the check stage, and without list size restrictions, is not in the first entries of the list. Thus, if the list size is limited, the check stage avoids wrong solutions, being successful, but it does not provide a combined map as a solution. In order to properly evaluate the list size influence, the ratio between the number of successful combined maps obtained and the total map combinations is also calculated and provided in the simulation results. Such a rate is a part of the total success rate.

The hexagon maps with a hexagon side length of 1 meter are used as user and prior maps in the simulations. The standard map matching method with the check stage is used for finding the map combinations with $NT = 3$ and 4 transmitters, and the list size is varied from 1 to 250 entries. Altogether, 105 simulations are performed per method configuration and per list size. As mentioned, both the success rate and the number of successful combined maps obtained out of the total number of map combinations, per method configuration and list size, are calculated.

Figures 4.7 (a) and (b) show the simulation results obtained by the standard method with the check stage as a function of the maximum list size, using $NT = 3$ and 4 transmitters, respectively. The success rate is depicted in blue, while the percentage of successfully combined maps obtained is drawn in dark orange. The difference of these two ratios corresponds to the percentage of wrong solutions avoided by the method with the check stage, without providing a combined map. As a reference, the success rate of the standard method without the check stage is drawn in black.

Simulation results show that even the usage of a short list has major advantages over the method without the check stage. For example, carrying out the check stage only once, i.e. fixing the list size at 1, considerably increases the robustness of the map matching through the avoidance of wrong map combinations. In this situation, the percentage of successful map combinations equals the success rate of the method without the check stage, since no more solutions are tried.

As the list size grows, the success rate slightly decreases. This is caused because more solutions are evaluated and, in some of them, ambiguities in the transmitters' visibility areas make the check stage to be unable to detect and avoid wrong solutions. Nevertheless, the list size growth also increases the number of successful combined maps obtained. On balance, the usage of larger list sizes is preferable.



(a) Standard method with check stage, $NT = 3$ (b) Standard method with check stage, $NT = 4$

Figure 4.7.- Simulation results of the standard method with the check stage as a function of the maximum list size. The results have been obtained after performing 105 simulations per configuration and hexagon size. The success rate and the percentage of successful combined maps obtained are drawn in blue and dark orange, respectively. The performance of the standard method without the check stage is depicted in black.

Apart from previous conclusions, the simulation results show that the list size of a method with the check stage can be limited to some value without decreasing its performance. In the particular configurations of the simulated standard method, the results show that the list size can be fixed to 30 and 24 entries, respectively, when using $NT = 3$ and 4 transmitters. For larger list sizes, the calculated rates remain constant and the success rate values are the ones already shown in Tables 4.1 and 4.2 for the same methods' configurations, where no list size restrictions were applied.

Hence, even though the number of times that the check stage is performed cannot be exactly determined, the maximum list size can be fixed without a performance loss. That maximum list size can be used as an upper bound in the calculation of the check stage complexity.

4.3.2.- Map Size Influence

In this subsection, the computational complexity of the check stage is obtained and evaluated through simulation results. The asymptotic behavior of the check stage is analyzed through the Big O notation, usually denoted as $\mathcal{O}$. This notation is used to express the worst-case scenario for a given algorithm by providing an upper bound for its growth rate.

As described in Subsection 3.3.2, the visibility mismatch ratio calculation requires to go through user and prior maps to look for their common hexagons and compare their visibility information. Operations inside the nested loop are independent from any algorithm input, and assumed as time-constant.

The check stage also needs to perform some operations for each associated transmitter. However, these operations are omitted in the check stage asymptotic behavior calculation because the number of associated transmitters is assumed to be constant.

With this, the complexity of a single run of the check stage can be delimited by $\mathcal{O}\left(N_{\text{hex,u}} \cdot N_{\text{hex,p}}\right)$, being $N_{\text{hex,u}}$ and $N_{\text{hex,p}}$ the number of hexagons of the user and prior maps, respectively. In fact, this asymptotic behavior also includes the combined map obtainment: the map combination is performed in the same nested loop for avoiding a large number of operations. Then, if the visibility information of the user and prior maps mismatch, the combined map is discarded.

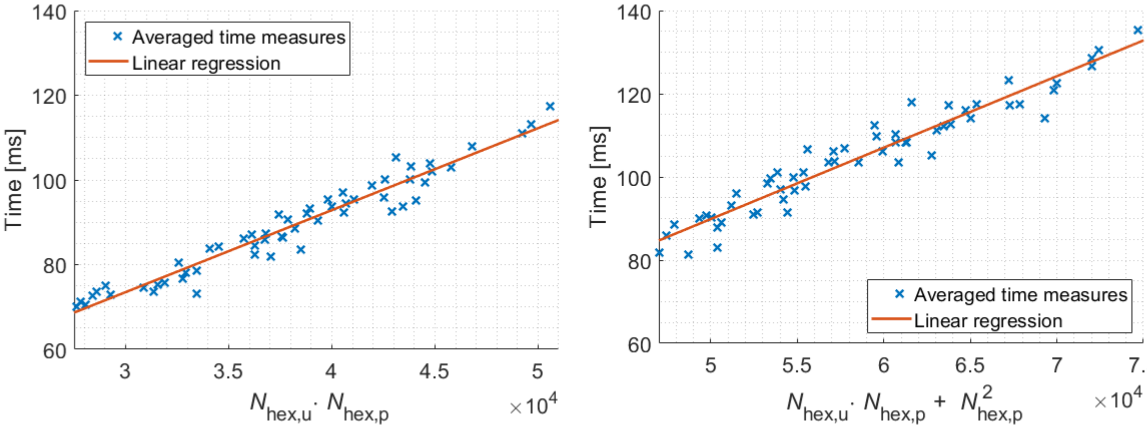
The asymptotic behavior of a single run of the check stage has been already calculated. However, if the visibility information of the maps that are combined differs, the prior map needs to be transformed again, using this time the parameters of the next entry of the list. Therefore, the prior map transformation complexity should be included as a part of the check stage complexity.

The prior map transformation requires to rotate and translate both the transmitters and the hexagon map. Again, the number of transmitters is assumed to be constant, being omitted in the asymptotic behavior calculation. In the hexagon map transformation, the center of each hexagon is rotated and translated over a fixed grid. The information of the original hexagon is incorporated into the hexagon of the fixed grid where the center falls after being transformed. Two hexagons of the initial prior map can fall into the same hexagon of the fixed grid; thus, a neested loop needs to be used to check if the current hexagon of the prior map that is transformed already exists in the transformed map.

The size of the outer loop is fixed to the number of hexagons of the prior map $N_{\text{hex,p}}$, while the size of the inner loop corresponds to the number of hexagons of the transformed map, which is initially empty but grows during the process. In the worst case, every hexagon of the prior map, after being transformed, creates a new hexagon in the transformed map, augmenting its size in 1 hexagon per iteration and ending in a $N_{\text{hex,p}}$ size. The number of total iterations in this situation is $N_{\text{hex,p}}^2/2 - N_{\text{hex,p}}/2$. Using the Big O notation, the complexity of the map transformation is given by $\mathcal{O}\left(N_{\text{hex,p}}^2\right)$. Finally, the complexity of a single run of the check stage including the prior map transformation and the map combination results in $\mathcal{O}\left(N_{\text{hex,u}} \cdot N_{\text{hex,p}} + N_{\text{hex,p}}^2\right)$.

In order to verify the asymptotic behavior of the check stage, including and not including the prior map transformation, the corresponding elapsed times of every check stage iteration have been stored for all performed simulations of the standard method with $NT = 3$. The usage of the 15 initial maps with 3 hexagon sizes as user and prior maps allow to perform 315 simulations, with different number of hexagons per map. These simulations have been repeated 20 times, and in each one the check stage is performed more than once. These two facts allow to average the measured times.

The measured times of the check stage including the map combination are represented in Figure 4.8 (a) as a function of the product $N_{\text{hex,u}} \cdot N_{\text{hex,p}}$. In Figure 4.8 (b), the averaged elapsed times of the check stage incorporating both the map combination and the prior map transformation are drawn as a function of $N_{\text{hex,u}} \cdot N_{\text{hex,p}} + N^2_{\text{hex,p}}$. In order to evaluate the asymptotic behavior of the check stage, in both cases the elapsed times are represented as a function of high values of the inputs $N_{\text{hex,u}}$ and $N_{\text{hex,p}}$.



(a) Not including prior map transformation

(b) Including prior map transformation

Figure 4.8.- Check stage complexity behavior obtained through simulations for the standard method with $NT = 3$. In (a), the prior map transformation is not taken into account in the measured times. In (b), the whole check stage time was measured, including prior map transformation, visibility mismatch ratio calculation and evaluation, and map combination. A linear regression is included in both illustrations to emphasize their behavior in relation to the x axes.

Both figures show, for the represented high values of the check stage algorithm inputs $N_{\text{hex,u}}$ and $N_{\text{hex,p}}$, a linear behavior in relation to the represented x axes. This fact confirms the theoretically estimated asymptotic behavior of the check stage. As shown in Figure 4.8 (a), the computational cost of the check stage itself, including the map combination process, behaves as $\mathcal{O}\left(N_{\text{hex,u}} \cdot N_{\text{hex,p}}\right)$. The inclusion of the prior map transformation, needed for calculating the visibility mismatch ratio and obtaining the combined map, makes that behavior change to $\mathcal{O}\left(N_{\text{hex,u}} \cdot N_{\text{hex,p}} + N^2_{\text{hex,p}}\right)$.

The obtained behavior corresponds to a single run of the check stage. Assuming that the list size studied in the previous subsection is limited to a certain value N_{list}, the prior map transformation, visibility mismatch ratio calculation and map combination would be performed, in the worst case, N_{list} times. Thus, the global computational complexity is given by $\mathcal{O}\left(N_{\text{list}}(N_{\text{hex,u}} \cdot N_{\text{hex,p}} + N^2_{\text{hex,p}})\right)$.

The number of hexagons of a map is closely related to the hexagon size: the smaller the hexagon size, the higher the number of hexagons needed to cover the same area. Thus, the obtained asymptotic behavior confirms the trade-off relation between the map matching method with check stage performance, analyzed in Section 4.2 in terms of success rate, and computational complexity.

5.- Conclusions and Outlook

For concluding the present Master's thesis, a summary of the performed work and the main outcomes derived from such work are included in Section 5.1. Finally, a generic outlook is provided in Section 5.2.

5.1.- Conclusions

The Channel-SLAM position accuracy can be improved if prior information of the scenario, i.e. transmitters' locations, is obtained. In multiple user scenarios, map estimations from other users can be obtained through crowd-sourcing. Nevertheless, each user creates its map in a local coordinate system. In order to use the maps from other users, transformation parameters need to be estimated in the map matching process.

Classical map matching methods are only based on the transmitters' positions, since those positions were the only information contained in the Channel-SLAM maps. However, symmetries in the transmitters' locations may lead to errors in the map matching process. A wrong map combination can introduce false information in the user map, which may lead to a divergence of the Channel-SLAM algorithm. Furthermore, the erroneous map can be shared, in turn, with other users, and the error can be propagated.

In the present Master's thesis, the transmitters' visibility areas have been used to improve the robustness of the map matching methods. For this purpose, three different tasks have been performed.

Firstly, a suitable way to store the transmitters' visibility information during the Channel-SLAM operation has been developed. A grid map formed by hexagons is used for this purpose. With this, the classical Channel-SLAM map has been changed from a feature map to a hybrid map. Now, along with the transmitters' estimated positions, a hexagon grid map with transmitters' visibility information is included in the Channel-SLAM map.

Secondly, a visibility metric has been developed to make use of the transmitters' visibility information stored in the Channel-SLAM maps. This visibility metric allows to compare two maps in terms of visibility information, given a set of correspondences between the transmitters of the user and the prior maps. The developed visibility metric considers the reliability of the transmitters' visibility information that the maps contain.

Finally, a check stage that makes use of the visibility metric has been created to improve the performance of the Channel-SLAM map matching methods. This check stage is compatible with every already developed map matching method based on the transmitters' positions.

Although the developed visibility map estimation, metric and check stage have been conceived to specifically be used in Channel-SLAM, they can be applied to other positioning methods based on anchors or landmarks that are visible only in certain areas.

In order to prove the usefulness of the developed techniques based on visibility information in the map matching process, a simplified version of Channel-SLAM as well as two configurable map matching methods based on transmitters' positions have been implemented in a simulation software. One of the map matching methods, used for obtaining the greater part of the simulation results and so-called standard method in this thesis, has been extracted from [7]. The other map matching method, named rotation method, has been designed from scratch. For its part, the performed implementation of Channel-SLAM is based on [4] and [30].

The developed visibility map estimation has been included in the Channel-SLAM implementation to be able to generate Channel-SLAM hybrid maps by means of simulations. A different version of the standard and rotation methods has been created after including the created check stage based on visibility information in their final part. Thus, the performance of the methods with and without the check stage can be compared.

A nearly symmetric scenario, which leads to symmetries in the transmitters' positions, has been created to hold the performance comparisons. This scenario has been selected due to the performance of the classical map matching methods only based on transmitters' locations is expected to be very poor. Then, it can be seen if the inclusion of the developed check stage can improve such performance.

In addition, 14 user tracks, with different length and geometry, have been created to perform the Channel-SLAM simulations and obtain different maps. In each Channel-SLAM simulation, three different hexagon sizes have been used to create the maps. Thus, three maps with different hexagon sizes are obtained per simulation, which allow to carry out fair map matching performance comparisons depending on the hexagon size. A true map of the scenario has also been created for each hexagon size. Altogether, 45 maps are available to evaluate the performance of the map matching methods, 15 per hexagon size.

The standard and rotation map matching methods with and without the check stage, and with different configurations, have been used to perform map combinations using the created maps. Analyzing the obtained results, a success rate can be obtained per method configuration.

Comparing the obtained results by the methods with and without the check stage, it has been shown that the check stage inclusion based on visibility can considerably improve the performance of the classical methods. For every implemented method, and with every possible configuration, the check stage inclusion makes the method to succeed in a considerably larger amount of cases. Thus, the simulations have shown that the inclusion of the developed check stage can solve ambiguities caused by the symmetries in the transmitters' positions and increase the robustness of the classical map matching methods, which was the main objective of this thesis.

Through the comparison of the obtained results using the map matching methods with the check stage for different hexagon sizes of the maps, it has been obtained, for the evaluated sizes, that a size reduction provides better results. This is due to the visibility information of the scenario is better suited. Despite this, it also has been obtained that even the usage of larger hexagons greatly increases the robustness of the map matching methods without check stage.

Along with the performance evaluation, a computational complexity analysis has been theoretically and experimentally carried out. First, it has been found that a reduction of the parameter list size of the check stage can be made without deteriorating the success rate of the map matching method. Thus, the established maximum list size can be used as an upper bound to calculate the computational complexity of the check stage. In addition, it has been shown that the usage of a short parameter list is not preferable, but it still increases the robustness of the classical map matching methods significantly.

Second, the computational complexity of the check stage has been theoretically obtained, and experimentally verified through simulation results. In summary, the check stage computational complexity grows linearly with the size of the map matching parameter list, the number of hexagons of the prior map, and the sum of the number of hexagons of the user and the prior maps.

The hexagon size determines the number of hexagons that a map needs to cover a certain area. Thus, the obtained complexity results prove, along with the success rate results mentioned before, that the hexagon size controls a trade-off relation between the map matching success performance and computational complexity.

5.2.- Outlook

It should be underlined that even when the check stage inclusion does not modify the solution obtained by the classical method only based on the transmitters' locations, it still provides a visibility metric which indicates how good, in visibility terms, the map combination has been. This metric can be used in further applications as a reliability metric of the combined map.

For example, in case a user receives previously combined maps from different users, the visibility metrics of the received maps can determine which one should be used in the map matching. Another application of the developed metric is the performance comparison of map matching methods based on the transmitters' locations.

Finally, the visibility areas now included in the Channel-SLAM maps can be used, along with the transmitters' positions and the occupancy probability of the hexagons, to rebuild the scenario geometry. Furthermore, the visibility information can also be useful in the data association process performed in the first stage of Channel-SLAM, which consists in finding correspondences between transmitters and measurements.

Thus, the visibility information inclusion in Channel-SLAM not only improves the map matching performance, but also opens a wide variety of future research directions.

Bibliography

[1] G. Dedes and A. G. Dempster. Indoor GPS Positioning - Challenges and Opportunities. In *IEEE 62nd Vehicular Technology Conference Fall*, volume 1, pages 412–415, September 2005.

[2] C. Gentner and T. Jost. Indoor Positioning using Time Difference of Arrival between Multipath Components. In *International Conference on Indoor Positioning and Indoor Navigation*, pages 1–10, October 2013.

[3] C. Gentner, B. Ma, M. Ulmschneider, T. Jost, and A. Dammann. Simultaneous Localization and Mapping in Multipath Environments. In *IEEE/ION Position, Location and Navigation Symposium (PLANS)*, pages 807–815, April 2016.

[4] C. Gentner, T. Jost, W. Wang, S. Zhang, A. Dammann, and U. Fiebig. Multipath Assisted Positioning with Simultaneous Localization and Mapping. *IEEE Transactions on Wireless Communications*, 15(9):6104–6117, September 2016.

[5] H. Durrant-Whyte and T. Bailey. Simultaneous Localization and Mapping: Part I. *IEEE Robotics Automation Magazine*, 13(2):99–110, June 2006.

[6] M. Ulmschneider and C. Gentner. Improving Maps of Physical and Virtual Radio Transmitters. In *ION GNSS+*, pages 3367–3373, September 2018.

[7] M. Ulmschneider, D. C. Luz, and C. Gentner. Exchanging Transmitter Maps in Multipath Assisted Positioning. In *IEEE/ION Position, Location and Navigation Symposium (PLANS)*, pages 1020–1025, April 2018.

[8] P. Robertson, M. Garcia Puyol, and M. Angermann. Collaborative Pedestrian Mapping of Buildings Using Inertial Sensors and FootSLAM. In *ION GNSS*, September 2011.

[9] J. J. Leonard and H. F. Durrant-Whyte. Simultaneous Map Building and Localization for an Autonomous Mobile Robot. In *IEEE/RSJ International Workshop on Intelligent Robots and Systems*, volume 3, pages 1442–1447, November 1991.

[10] J. Tardos, J. Neira, P. M. Newman, and J. J. Leonard. Robust Mapping and Localization in Indoor Environments Using Sonar Data. *The International Journal of Robotics Research*, 21, April 2002.

[11] L. Kleeman. Advanced Sonar and Odometry Error Modeling for Simultaneous Localisation and Map Building. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, pages 699–704, October 2003.

[12] A. J. Davison. Real-Time Simultaneous Localisation and Mapping with a Single Camera. In *Proceedings Ninth IEEE International Conference on Computer Vision*, pages 1403–1410, October 2003.

[13] A. J. Davison, I. D. Reid, N. Molton, and O. Stasse. MonoSLAM: Real-Time Single Camera SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29:1052–1067, June 2007.

[14] G. Klein and D. Murray. Parallel Tracking and Mapping for Small AR Workspaces. In *6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 225–234, November 2007.

[15] M. Kaess and F. Dellaert. Probabilistic Structure Matching for Visual SLAM with a Multi-Camera Rig. *Computer Vision and Image Understanding*, 114(2):286–296, February 2010. Special issue on Omnidirectional Vision, Camera Networks and Non-conventional Cameras.

[16] F. Fang, X. Ma, and X. Dai. A Multi-Sensor Fusion SLAM Approach for Mobile Robots. In *IEEE International Conference Mechatronics and Automation*, volume 4, pages 1837–1841, July 2005.

[17] P. Meissner, E. Leitinger, M. Fröhle, and K. Witrisal. Accurate and Robust Indoor Localization Systems using Ultra-wideband Signals. *CoRR*, abs/1304.7928, April 2013.

[18] P. Setlur, G. E. Smith, F. Ahmad, and M. G. Amin. Target Localization with a Single Sensor via Multipath Exploitation. *IEEE Transactions on Aerospace and Electronic Systems*, 48(3):1996–2014, July 2012.

[19] A. O'connor, P. Setlur, and N. Devroye. Single-Sensor RF Emitter Localization based on Multipath Exploitation. *IEEE Transactions on Aerospace and Electronic Systems*, 51(3):1635–1651, July 2015.

[20] B. Ferris, D. Fox, and N. Lawrence. WiFi-SLAM Using Gaussian Process Latent Variable Models. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, IJCAI'07, pages 2480–2485, San Francisco, CA, USA, January 2007.

[21] J. Huang, D. Millman, M. Quigley, D. Stavens, S. Thrun, and A. Aggarwal. Efficient, Generalized Indoor WiFi GraphSLAM. In *IEEE International Conference on Robotics and Automation*, pages 1038–1043, May 2011.

[22] S. Saeedi, M. Trentini, M. Seto, and H. Li. Multiple-Robot Simultaneous Localization and Mapping: A Review. *Journal of Field Robotics*, 33(1):3–46, January 2016.

[23] A. Elfes. Using Occupancy Grids for Mobile Robot Perception and Navigation. *Computer*, 22(6):46–57, June 1989.

[24] S. Kay. *Fundamentals of Statistical Signal Processing: Estimation Theory.* Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993.

[25] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A Tutorial on Particle Filters for Online Nonlinear/non-Gaussian Bayesian Tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, February 2002.

[26] N. J. Gordon, D. J. Salmond, and A. F. M. Smith. Novel Approach to Nonlinear/Non-Gaussian Bayesian State Estimation. *Radar and Signal Processing, IEE Proceedings F*, 140:107–113, May 1993.

[27] A. Kong, J. S. Liu, and W. H. Wong. Sequential Imputations and Bayesian Missing Data Problems. *Journal of the American Statistical Association*, 89(425):278–288, March 1994.

[28] T. Jost, W. Wang, U. Fiebig, and F. Pérez-Fontán. Detection and Tracking of Mobile Propagation Channel Paths. *IEEE Transactions on Antennas and Propagation*, 60:4875–4883, October 2012.

[29] W. Wang, T. Jost, and A. Dammann. Estimation and Modelling of NLoS Time-Variant Multipath for Localization Channel Model in Mobile Radios. In *IEEE Global Telecommunications Conference GLOBECOM*, pages 1–6, December 2010.

[30] M. Ulmschneider, C. Gentner, T. Jost, and A. Dammann. Rao-Blackwellized Gaussian Sum Particle Filtering for Multipath Assisted Positioning. *Journal of Electrical and Computer Engineering*, April 2018.

[31] G. Casella and C. P. Robert. Rao-Blackwellisation of Sampling Schemes. *Biometrika*, 83:81–94, March 1996.

[32] A. Doucet, N. Freitas, K. P. Murphy, and S. J. Russell. Rao-Blackwellised Particle Filtering for Dynamic Bayesian Networks. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, UAI '00, pages 176–183, San Francisco, CA, USA, 2000.