A Composite Beacon Initialization for EKF Range-Only SLAM

Lionel Génevé, Olivier Kermorgant, Edouard Laroche

A Composite Beacon Initialization for EKF Range-Only SLAM

Lionel Génevé*, Olivier Kermorgant*, Édouard Laroche*
*ICube Laboratory, University of Strasbourg-CNRS,
Strasbourg, France
{l.geneve, kermorgant, laroche}@unistra.fr

***Abstract*— Accurately localize a mobile vehicle with an easy and quickly deployable system can be very useful for many applications. Herein we present an EKF-SLAM algorithm which allows using radio frequency (RF) beacons without any prior knowledge of their location. As RF beacons provide only range information, recovering their positions is not an easy task. For this range-only SLAM case, a new procedure to instantiate the beacons in the filter is proposed. The method uses two range measurements from different robot's positions to initialize two hypotheses for the beacon's location, which are then integrated in the filter as a Gaussian mixture. This approach is evaluated and compared to other initialization techniques in simulation and with a real dataset. The results show that our approach performs as well as the other existing methods for both trajectory and map errors with a low computational cost.**

***Index Terms*— SLAM, Range-only, EKF, mobile robot**

I. Introduction

For a mobile vehicle, the ability to localize itself in the environment is an essential requirement to perform more advanced tasks such as path following or reaching a target. In the case where the map of the environment is not available, the autonomous vehicle has to deal with the mapping problem at the same time. This approach is called simultaneous localization and mapping (SLAM) and is a classical problem in mobile robotics. One of the first solution [1] uses an extended Kalman filter (EKF) to estimate the robot's pose and the beacons' positions. This approach relies on Gaussian assumptions for the noises of the process and measurement models to represent the state's belief as a Gaussian distribution. It is necessary to have a good estimation of the state to linearize the equations. Despite a quadratic complexity which scales with the number of beacons, the EKF is still very efficient when this number is kept low compared to other algorithms as particle filters or optimization based techniques. So here, we decided to use the EKF framework as the base for our algorithm.

For feature-based SLAM, it can be difficult to extract relevant features or to identify them. The data association problem, which consists in associating a measurement to the corresponding feature adds some complexity in the implementation. Adding easily identifiable beacons in the environment can get around this problem. But, in some applications it can be constraining and difficult to place these beacons. Keeping the number of beacons as low as possible is this necessary.

Two types of information can potentially be retrieved from the beacons: bearing and range with respect to the robot. As we focus on developing RF beacons with time-of-flight (ToF) or received signal strength indicator (RSSI) [2], this article deals with the second case: range-only SLAM (RO-SLAM). The benefits of using RF beacons are twofold: the correspondence problem can be easily solved by sending the ID of the beacon in the message, and it is possible to have non-line-of-sight ranging. On the other hand, the measurement equation is highly nonlinear, and the exact beacon's position cannot be recovered with only one measurement. In RO-SLAM, the main problem comes from the partial observability. This means that several measurements are necessary to fully initialize a beacon's position in the state vector of the filter. In this paper, a composite method to initialize the RF beacons within the EKF framework is presented. An Euclidean parameterization is used for both the robot's pose and beacons' positions to have a minimal size for the state vector. We also use a short delayed approach using only two range measurements to initialize the beacon's position in the filter with a two-hypothesis Gaussian mixture. The two hypotheses are tracked and updated with the next measurements until one of them is pruned.

The rest of this paper is organized as follows. First, the existing methods for RO-SLAM are presented. Then, in section III, our approach is detailed with the focus on the initialization scheme. In section IV, our method is tested and compared to other approaches in simulation and with a real dataset. Finally, section V concludes on the approach, and some limitations along with further research are exposed.

II. Related work

For the SLAM problem with only range information, it is very challenging to initialize totally the beacon's position in the filter with one measurement. Several techniques already exist to perform this initialization. They can be classified in delayed or undelayed methods.

In the delayed methods, the range measurements are gathered until enough information is available to localize the beacon, and insert its position into the filter. With trilateration [3], three range measurements recorded at different robot's positions are used to compute the intersection of three circles provided that the centers do not lie on a same line. In spite of its simplicity, this method is very sensitive to noise, and it is useful to employ more than three range measurements. Leonard et al. [4] proposed a technique to initialize different kinds of features from range measurements by incorporating

the associated robot's poses in the filter until the initialization can be performed efficiently. The accumulation of measurements should improves the initial estimate of the beacon. But, if we wait too long, the odometry error of the robot could grow too much to correctly perform the initialization.

Olson et al. [5] tried to cope with this problem by using a 2D probabilistic grid of the world with an accumulator voting scheme. Each pair of range measurement provides two possible solutions for the beacon's position. The value of the grid for these two positions are incremented by one each time a new measurement ties in. Finding the cell in the grid with the higher ratio of votes allows initializing the beacon's position.

Another solution is to use a particle filter as in [6] within an approach similar to FastSLAM. When the first range measurement coming from a beacon is available, we only know that the beacon's position is located on a circle centered on the robot's position, with a radius equal to the range measurement. This information is used to represent the possible beacons' positions by a probability density approximated by the particles of the filter. At the begining, the probability density is uniformly distributed around the circle. By accumulating other range measurements from the beacon, each particle's weight is updated according to the probability that the range measurement comes from this particle. This probability density should converge to a single location, from which we can compute a mean and a variance that will be used to initialize the beacon.

For the undelayed methods, since the first range measurement, the hypotheses of the beacon's position are inserted in the filter. Generally the beacon's state is represented with a multi-modal distribution. When new measurements are available, bad hypotheses are pruned accordingly until a single hypothesis remains. The main advantage of these methods is that since the begining, the information provided by the range measurements can be used to update the filter.

One approach is to use a mixture of Gaussians as in [7] and [8]. In their work, at least four variables are necessary to represent a beacon (in 2D): the robot's position of the first range measurement as the origin of the circle, the associated range measurement and the angle. A weighted product of k Gaussians is used to represent the angular uncertainty on the circle. As each hypothesis comes with its own mean and variance, the usual KF update steps can be applied to each of them. The weights of each hypothesis evolve according to the likelihood that a range measurement satisfies this hypothesis. When a weight falls below a threshold, the corresponding hypothesis is removed from the filter. This method provides good performances but at a high computational cost, increasing with the number of hypotheses.

Another technique, presented in [9] is the Relative Over Parametrization (ROP). The beacons' positions and the robot's pose are expressed in polar coordinates. This formulation models better ring and circle shapes encountered in RO-SLAM, and thus the nonlinearities and multi-modal problems. The beacon's positions are over-parameterized by using the polar coordinates, plus the origin of the polar frame in Cartesian coordinates, leading to an increased state vector. The first range measurement from a beacon is used to add the beacon's position in the filter, by associating a very large variance along the angle to get a nearly uniform probability between $]-\pi, \pi]$. After the second measurement, the intersection of two circles gives two points and thus two hypotheses. The uniform probability is split into two hypotheses as for a multi-hypotheses scheme, until it converges to a unique unimodal estimate.

RO-SLAM was also implemented with other frameworks than the KF. A Sparse Extended Information Filter (SEIF) version was detailed in [10] which exploits the inter-beacon measurements, and the computational capabilities of the beacons to distribute the update step of the filter among them. Optimization techniques were employed as in [11], with ISAM in [12] or GraphSLAM in [13]. In [14], a spectral learning approach was presented. The principle is to construct a matrix with the squared range measurements gathered at each time step. This matrix is factored in two matrices: one with the robot's positions and the other with the beacon's positions. A Singular Value Decomposition (SVD) is then used to recover the robot and the beacons' positions but up to a linear transformation. Recently, Lourenço et al. [15], addressed the problem of RO-SLAM with a globally exponentially stable (GES) filter. They combined the robocentric framework introduced in [16], and state augmentation inspired by the source-localization algorithm in [17], to design the filter in a body-fixed frame, and obtain a linear time-varying formulation. They performed an observability analysis and established requirements on the vehicle's trajectory. The system is solved using a KF with GES error dynamics.

For RO-SLAM, the beacons must be initialized as soon as possible to benefit from their measurements. Moreover, to achieve a low complexity and good computational performances, it is desirable to keep a minimal parameterization of the state vector. The next section details an algorithm addressing these requierements.

III. Algorithm overview

In this section, the different steps of the proposed algorithm are detailed. The main idea is to incorporate the pose associated with the first range measurement in the state vector as in [4], and to wait the availability of a second measurement to initialize the beacon. Similar to [7], a Gaussian mixture is used to represent the beacon's position, but here with only two hypotheses and a cartesian representation. First, we start by introducing the system and the notations used throughout this paper.

A. System overview

1) Robot: We consider a differential drive mobile robot evolving in an outdoor planar world. Its pose in the world frame is defined by: $\mathbf{p} = (x, y, \theta)^T$. The vehicle is equiped with wheel encoders which give the traveled distance ΔU (in m), and the change of orientation $\Delta\Theta$ (in rad) between

two timestep. The corresponding odometry measurement is:

$$\mathbf{U} = \begin{pmatrix} \Delta U \\ \Delta\Theta \end{pmatrix} + \epsilon \tag{1}$$

where ϵ represents the white Gaussian noise on the traveled distance and change of orientation. In reality, other noises not modeled here can corrupt the measurements like wheels slippage or approximations of the vehicle geometry.

2) Beacons: The choice of RF beacons providing range measurements was made for the reasons explained in the introduction. As the system must be quickly and easily deployed, only a few beacons are used (6 in the simulation and 4 for the real dataset). We consider that N static and identifiable beacons B_i are installed in the world at unknown locations: $\mathbf{x}_i = (x_i, y_i)^T$. The range measurement equation between a beacon B_i and the robot is given by:

$$r_i = \sqrt{(x_i - x)^2 + (y_i - y)^2} + \nu_i \tag{2}$$

Where ν_i is the error on the distance, supposed to be the realization of a white Gaussian noise with a variance of σ_i^2. In practice, additional perturbations appear such as a bias proportional to the measured distance, and a positive bias caused by interferences or obstacles in the line-of-sight. These perturbations are not modeled here, but were simulated in our experiments.

3) Algorithm: When it is not possible to know the position of each beacon in the world frame, a SLAM algorithm is needed. Despite that the number of necessary beacons depends on the size and complexity of the environment, the large coverage zone of a RF beacon allows us to use only a small number of beacons. Thus, the quadratic complexity will be limited, allowing to use an EKF. Although it was stated in [9] that polar coordinates can be more accurate because they deal better with nonlinearities and multi-modal distributions, we use Cartesian coordinates for the robot's pose and beacons' positions to keep a minimal parameterization of the state vector.

The state vector at time k when all the beacons are fully initialized is of the form:

$$\mathbf{X}_k = (\mathbf{p}_k, \mathbf{x}_1, ..., \mathbf{x}_i, ..., \mathbf{x}_N)^T \tag{3}$$

with $\mathbf{p}_k = (x_k, y_k, \theta_k)^T$ being the pose of the robot at time k. For the case where there exists two hypotheses for a beacon, the state vector simply stores the coordinates of each hypothesis corresponding to the beacon: $(x_i^1, y_i^1, x_i^2, y_i^2)^T$. We now present the model for the prediction step.

B. Prediction step

The prediction only affects the state variables of the robot. The equations are derived from (1) and are given by:

$$\begin{cases} x_{k+1} = x_k + \Delta U \cos(\theta_k + \frac{\Delta\Theta}{2}) \\ y_{k+1} = y_k + \Delta U \sin(\theta_k + \frac{\Delta\Theta}{2}) \\ \theta_{k+1} = \theta_k + \Delta\Theta \end{cases} \tag{4}$$

We choose to define a noise proportional to the distance traveled and to the change in orientation. The EKF odometry measurement noise matrix thus yields:

$$\mathbf{Q} = \begin{pmatrix} k_{\Delta U}|\Delta U| & 0 \\ 0 & k_{\Delta\Theta}|\Delta\Theta| \end{pmatrix} \tag{5}$$

where $k_{\Delta U}$ and $k_{\Delta\Theta}$ are tuning parameters representing the confidence we have with respect to the linear and angular displacements.

In our implementation we also check at each iteration if it is necessary to include an old robot's pose $\mathbf{p}_{old} = (x_{old}, y_{old}, \theta_{old})^T$ in the state of the filter, to perform the state augmentation described in [4]. The uncertainties associated to the old poses are kept and used to initialize the beacon's position and variance when a new measurement is available. The augmented state vector is:

$$\mathbf{X}_k = (\mathbf{p}_k, \mathbf{p}_{old}, \mathbf{x}_1, ..., \mathbf{x}_i, ..., \mathbf{x}_N)^T \tag{6}$$

Once the first range measurement and its corresponding pose are recorded, the algorithm waits until a second range measurement is available to perform the initialization. This procedure to initialize is presented in the next section.

C. Beacon initialization

A good initial estimate for the beacons' positions is crucial for the KF to be able to converge towards the true location, otherwise the estimation will be erroneous. We noticed that with a multi-hypothesis approach the number of possible locations for a beacon quickly drops to two and then takes more time to converge to a unique solution. This motivates our choice to initialize the beacon's position in the state with only two range measurements. By computing the intersection of two circles, it is expected to get two solutions that are used as two hypotheses in a Gaussian mixture.

The state vector $\mathbf{X}_k$ of the filter is initially composed of the robot's pose $\mathbf{p}_k = (x_k, y_k, \theta_k)^T$ only. When the first range r_i from the beacon B_i is measured, the corresponding robot's pose is incorporated into the filter as in [4]. The associated range measurement is also saved. Once a second measurement is available, a first check is made that the Euclidean distance d_1 between the current pose and the old pose from the first measurement is greater than a fixed threshold t_1 (set proportional to σ_i in our experiments), where:

$$d_1 = \sqrt{(x_k - x_{old})^2 + (y_k - y_{old})^2} \tag{7}$$

If it is the case, we solve a second order equation from the intersection of two circles:

$$\begin{cases} (x_{old} - x_i)^2 + (y_{old} - y_i)^2 = r_{old}^2 \\ (x_k - x_i)^2 + (y_k - y_i)^2 = r_i^2 \end{cases} \tag{8}$$

The centers of the circles are the robot's positions, and the radius of each circle corresponds to the range measurement. One, two or no solutions are possible.

In the case where the two circles do not intersect, the algorithm waits for another range measurement to perform the initilization. If one solution is found, it is directly inserted into the state vector. If two solutions are obtained, they are instanciated in the filter as the hypotheses of a Gaussian mixture as in [7], but with the difference that here we keep

a Cartesian representation. The weights of each hypothesis are set to 0.5 so their sum is 1. The covariance matrices associated with each hypothesis are computed as in [4]. The new covariance matrix of the estimated state after inserting the two beacons is:

$$\mathbf{P}_k = \begin{pmatrix} \mathbf{P}_k & \mathbf{P}_k\mathbf{G}_x^T & \mathbf{P}_k\mathbf{G}_x^T \\ \mathbf{G}_x\mathbf{P}_k & \mathbf{P}_1 & \mathbf{0} \\ \mathbf{G}_x\mathbf{P}_k & \mathbf{0} & \mathbf{P}_2 \end{pmatrix} \tag{9}$$

with $\mathbf{P}_i = \mathbf{G}_x\mathbf{P}_k\mathbf{G}_x^T + \mathbf{G}_z\mathbf{R}\mathbf{G}_z^T$, $i = 1, 2$ and $\mathbf{R} = \mathrm{diag}(\sigma_\mathrm{i}^2, \sigma_\mathrm{i}^2)$. $\mathbf{G}_x$ and $\mathbf{G}_z$, are computed according to $\mathbf{G}_x = -\mathbf{H}_y^{-1}\mathbf{H}_x$ and $\mathbf{G}_z = \mathbf{H}_y^{-1}$. Here, $\mathbf{H}_x$ and $\mathbf{H}_y$ are the derivative of the left hand side of (8) with respect to the state and the beacon respectively. In our case:

$$\mathbf{H}_x = \begin{pmatrix} \mathbf{H}_{x,1} & \mathbf{0}_{2\times2} & \dots & \mathbf{0}_{2\times2} & \mathbf{H}_{x,2} & \mathbf{0}_{2\times2} & \dots \end{pmatrix} \tag{10}$$

$$\mathbf{H}_{x,1} = \begin{pmatrix} \frac{-2(x_i - x_{old})}{r_{old}} & \frac{-2(y_i - y_{old})}{r_{old}} \\ 0 & 0 \end{pmatrix} \tag{11}$$

$$\mathbf{H}_{x,2} = \begin{pmatrix} 0 & 0 \\ \frac{-2(x_i - x_k)}{r_i} & \frac{-2(y_i - y_k)}{r_i} \end{pmatrix} \tag{12}$$

$$\mathbf{H}_y = \begin{pmatrix} \frac{2(x_i - x_{old})}{r_{old}} & \frac{2(y_i - y_{old})}{r_{old}} \\ \frac{2(x_i - x_k)}{r_i} & \frac{2(y_i - y_k)}{r_i} \end{pmatrix} \tag{13}$$

If no more range measurements is associated with the old pose in the state filter, it is deleted. The steps are summarized in Algorithm 1. If the noise on the range measurements increases with the distance, it is also possible to use another threshold to discard abnormally high distances in the initialization process.

Input : $\mathbf{X}_k, \mathbf{P}_k, r_i, B_i, \sigma_i$
Output: Augmented $\mathbf{X}_k$ and $\mathbf{P}_k$

if B_i *observed for the first time* **then**
  Add current robot's pose in $\mathbf{X}_k$ (6) ;
  Update $\mathbf{P}_k$;
  Store the associated r_i and B_i ;
else
  Search for the old pose $\mathbf{p}_{old}$ and associated range measurement r_{old} corresponding to B_i ;
  Compute d_1 according to (7) ;
  if $d_1 > t_1$ **then**
    Solve (8) to get n_H valid hypotheses ;
    if $n_H > 0$ **then**
      for $j \leftarrow 1$ **to** n_H **do**
        Initialize the weights: $w_j = \frac{1}{n_H}$;
        Augment $\mathbf{X}_k$ with: (x_i^j, y_i^j) ;
        Compute $\mathbf{G}_x$ and $\mathbf{G}_z$ according to (10),(11),(12),(13) ;
        Update $\mathbf{P}_k$ using (9) and σ_i ;
      end
      If necessary, remove $\mathbf{p}_{old}$ from $\mathbf{X}_k$;
    end
  end
end

Algorithm 1: Beacon initialization algorithm

D. Update step and pruning

1) Update step: When the beacon's position has converged to a unique solution, a Mahalanobis test is performed before the update to reject range measurements too far from the prediction. The prediction is computed according to (2), whereas the covariance matrix for the Mahalanobis distance is computed using the innovation covariance matrix $\mathbf{S}$ of the filter. Computing $\mathbf{S}$ requires the Jacobian of the range measurement function, $\mathbf{H}$, which is given by:

$$\mathbf{H} = \begin{pmatrix} \frac{-dx}{r_i} & \frac{-dy}{r_i} & 0 & \dots & 0 & \frac{dx}{r_i} & \frac{dy}{r_i} & 0 & \dots \end{pmatrix} \tag{14}$$

with $dx = x_i - x_k$, and $dy = y_i - y_k$.

The threshold t_2 for the Mahalanobis test is set proportional to the range measurement noise σ_i. The advantage of the Mahalanobis test is twofold. This test increases the robustness of the filter against spurious measurements. And, by recording for each beacon the percentage of accepted measures, we get an indicator of the beacon's relevance at the end of the algorithm. A low percentage could indicate a bad initialization of the beacon. When there is still two hypotheses for the beacon, the weights of each hypothesis are updated by computing the likelihood l that the range measurement comes from this hypothesis as described in [7]:

$$l = p(r_i|\mathbf{p}_k, \mathbf{p}_i^j) = \frac{1}{\sqrt{2\pi\sigma_i^2}} \exp -\frac{(r_{pred} - r_i)^2}{2\sigma_i^2} \tag{15}$$

with $r_{pred} = \sqrt{(x_k - x_i^j)^2 + (y_k - y_i^j)^2}$.

In our experiments, we found it useful for updating the weights to attenuate the likelihood l by using $l^{\frac{1}{a}}$, with a an integer equal or greater than two. It slows down the evolution of the weights but allows to use more range measurements to take the decision of pruning or not an hypothesis. The update step is summarized in Algorithm 2.

2) Pruning: Two tests are realized to see if an hypothesis must be pruned. The first one concerns the case where an hypothesis becomes negligible compared to the other. To do this, the absolute difference between the weights of each hypothesis is computed. If this difference is higher than a threshold (set at 0.9 in our experiments), the hypothesis with the lower weight is deleted from the filter. In the second test, we try to merge the hypotheses that are too close from each other. For this purpose, the Euclidean distance between the two hypotheses is computed. If the distance falls below a threshold, proportional to σ_i, the hypotheses are merged into one. After removing every hypothesis for each beacon, the form of the state vector is the one shown in (3). This is a minimal parametrization to represent the robot's pose and the beacons' locations. In the next section, the validity of this approach is asserted and compared to other initialization techniques.

IV. Experiments

To test and compare our implementation we used two type of datasets. The first one comes from a simulation environment, whereas the second one is a publicly available experimental dataset presented in [18]. It is intended for RO-SLAM

Input : $\mathbf{X}_k$, $\mathbf{P}_k, r_i, B_i, \sigma_i$
Output: $\mathbf{X}_k$, $\mathbf{P}_k$

n_H = number of hypotheses for beacon B_i ;
if $n_H == 1$ **then**
  Compute r_{pred} ;
  Compute $\nu = r_i - r_{pred}$;
  Compute $\mathbf{H}$ and S ;
  /* Mahalanobis test */
  if $\nu^T S^{-1} \nu < t_2$ **then**
    Compute $\mathbf{X}$ and $\mathbf{P}$
  end
else
  for $j = 1$ **to** n_H **do**
    $r_j = \sqrt{(x_j - x_k)^2 + (y_j - y_k)^2}$;
    $l_j = \frac{1}{\sqrt{2\pi\sigma_i^2}} e^{\frac{-(r_i - r_j)^2}{2\sigma_i^2}}$;
    $l_j = l_j^{\frac{1}{a}}$;
  end
  for $j = 1$ **to** n_H **do**
    $\lambda_j = \frac{l_j}{\sum_j l_j}$;
    $\sigma_j^2 = \frac{\sigma_j^2}{\lambda_j}$;
    $\omega_j = \omega_j l_j$;
  end
  for $j = 1$ **to** n_H **do**
    $\omega_j = \frac{\omega_j}{\sum_j \omega_j}$;
  end
  for $j = 1$ **to** n_H **do**
    $\nu_j = r_i - r_j$;
    $S_j = \mathbf{H}_j \mathbf{P}_k \mathbf{H}_j^T + \sigma_i^2$;
    $\mathbf{K}_j = \mathbf{P}_k \mathbf{H}_j^T S_j^{-1}$;
    $\mathbf{X}_k = \mathbf{X}_k + \mathbf{K}_j \nu_j$;
    $\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_j \mathbf{H}_j) \mathbf{P}_k$;
  end
end

Algorithm 2: Update step for a range measurement coming from beacon B_i

and was already used in [12]. Our solution is implemented in two versions. The first one, named $compositeAug$, perfoms the state augmentation as described in section III-B. The second, named $composite$, stores the robot's positions without inserting them in the filter, and initializes the variance of the beacons' positions with a diagonal matrix whose values are proportional to σ_i (the proportional term was set to 10 in our experiments). For comparison, we also implemented other initialization techniques using the EKF framework:

- A trilateration approach, termed $trilat10$, described in [3], with 10 range measurements used to perform the initialization
- A particle filter approach ($partFilter360$), inspired by the work of [6], with 360 particles to represent the different hypotheses
- A mixture of Gaussians $gaussMixt8$ and $gaussMixt16$ adapted from [7] with 8 and 16 hypotheses

To compare the performances of each technique, we compute the root mean squared of the error (RMSE) between the estimated and the true values of the robot's trajectory and the beacons' positions. We also record the execution time[1] and the iteration at which all the beacons were initialized in the filter (endInit).

A. Simulation data

Fig. 1. Snapshot of the simulation environment in Gazebo

Using ROS [19] and the Gazebo simulator, we teleoperated the vehicle with a maximal speed of 2 km/h on a planar ground to obtain a trajectory. The total trajectory's length is 54 m. A number of $N = 6$ beacons were set in the environment. The range measurements between the robot and a beacon are acquired at a frequency of 10 Hz. Moreover, the maximum range of the beacons was set to 100 m. The perfect odometry and range measurements were recorded, and afterwards processed to add noise and modeling errors. We added white Gaussian noises with a variance of $1.11e^{-3}$ m^2 on the odometry and on the range measurements for the whole trajectory. No proportional bias to the measured distance was considered. Despite the presence of obstacles in our environment, we didn't modelize any positive bias when the range measurements are passing through them. The trajectory was cut in seven phases, each one representing a different type of noise. Phases 1, 3, 5, 7 are only affected by the Gaussian noise. In Phase 2, we added a systematic error in the odometry measurements by introducing a small difference of 1 mm between the radius of the left and right wheels. In Phase 4, we simulated a high random noise on the range measurements to modelize some perturbations or interferences. In Phase 6, we added wheels slippage by randomly increasing the measured speed of one or both of the wheels during a few seconds. The vehicle starts from a known location. Thus, we assume that the initial state vector: $\mathbf{X}_0 = (x_0, y_0, \theta_0)^T$, and its associated covariance matrix: $\mathbf{P}_0$ are perfectly known. The parameters $k_{\Delta U}$ and $k_{\Delta\Theta}$ tuned according to the odometry performances, are set to $1.0e^{-3}$ and $1.0e^{-1}$ respectively. The variance for the range measurement noise is set to $\sigma_i^2 = 1.11e^{-3}$ m^2.

1) RMSE on the robot's trajectory: The evolution of the error for the robot's trajectory is plotted in Fig. 2. During Phase 1, with only Gaussian noise, the algorithm performs the initialization of the beacons, thus the error grows. This error continues to increase in Phase 2 with the bias on the odometry. During Phase 3, the error is stabilized until Phase

[1]Command cputime of Matlab

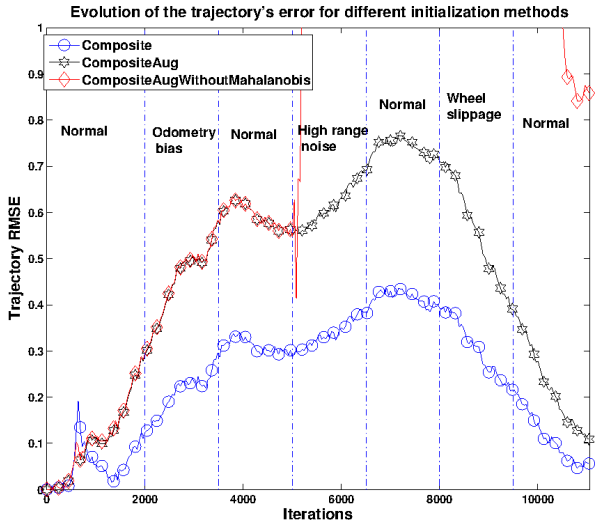

Fig. 2. RMS error of the robot's trajectory for the *composite* $(-o-)$, *compositeAug* $(-\star-)$ and *compositeAug* without the Mahalanobis test $(-\Diamond-)$ methods.

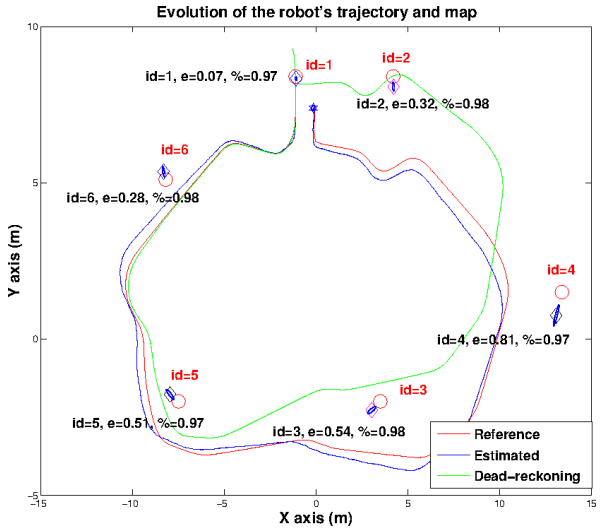

Fig. 3. Final robot's trajectory and beacons' positions of the simulation for the *compositeAug* method. e is the final error for the beacon, and $\%$ corresponds to the percentage of accepted range measurement with the Mahalanobis test. The RMSE for the final trajectory is 0.50 m.

4, where the high noise on the range measurements perturbates again the system. At this point, we see the effects of the Mahalanobis test to discard all the erroneous measurements. Finally, during the last phases, the error decreases because the estimates of the beacons' positions have converged and we observe that the wheels slippage is well compensated.

2) Comparison of the initialization techniques: In Table I, we summarized the results for the comparison between the different initialization techniques. One notices that the $trilat10$ approach, while accurate, takes a long time before initializing all the beacons in the filter. This is the fact that 10 measurements are needed for the initialization of a beacon, and because we imposed a minimal distance between two consecutive poses used in the initialization. $trilat10$ performs well because the odometry error is low, leading to good robot's pose estimations that do not compromise the trilateration. On the contrary, the Gaussian mixture is the fastest method to initialize the beacons, because they are inserted since the first measurement. But the accuracy with only 8 hypotheses is not very good and that 16 are needed to get better results but at the expense of a higher computation cost. The particle filter ($partFilter360$) approach performs very well for the trajectory but fails to accurately recover the beacons' positions. Our approach succeeds to correctly reconstruct the trajectory and the map. The final trajectory and beacons' positions of the *compositeAug* method are shown in Fig. 3. The results of *composite* and *compositeAug* are very similar, but with a small advantage for the *composite* method. We should expect the opposite because *compositeAug* keeps track of the uncertainties on the old robot's poses to propagate them on the initial beacon's position and thus have a closer estimate of the variance, but in this case *composite* performs better. Here again, when the Mahalanobis test is not used, the errors on the map and on the trajectory increase.

After evaluating the behavior of our approach on a trajectory with different types of noise and comparing the performances with other approaches, we also wanted to test the reliability of our approach on a real dataset.

Method	Trajectory RMSE (m)	Map RMSE (m)	Execution time (s)	endInit
$trilat10$	0.41	0.35	1.39	1222
$gaussMixt8$	2.44	11.68	5.72	11
$gaussMixt16$	0.32	0.57	6.3	11
$partFilter360$	0.17	5.52	1.45	764
composite	0.27	0.24	1.72	310
compositeAug	0.50	0.43	1.82	310
compositeAug w/o Mahalanobis test	2.97	3.07	1.7	310

TABLE I

COMPARISON OF DIFFERENT INITIALIZATION TECHNIQUES FOR THE SIMULATION ENVIRONMENT

B. Plaza1 dataset

This dataset provided by Djugash et al. in [18], was realized with a mobile robot in an outdoor environment by collecting range measurements with time-of-flight UWB (Ultra Wide Band) RF beacons. The variance σ_i^2 of the range measurements is set to 0.5 m^2 according to the description of the system. The robot is equiped with wheel encoders and a gyroscope to provide odometry measurements. As in the previous case, we set the noise associated to the odometry proportional to the traveled distance. The values corresponding to $k_{\Delta U}$ and $k_{\Delta\Theta}$ are set to $1.7e^{-5}$ and $1.0e^{-8}$ respectively. A ground truth from a 2-cm-accuracy DGPS is also provided for the trajectory of the robot and the beacons' positions.

The results for the comparison of the different initialization methods are presented in Table II. We added in our implementation a threshold to discard measurements greater than 30 m for the initialization process as explained in III-C. It explains why, the times when all the beacons are initialized in the filter are so delayed. Whereas the errors on the trajectory are close from one method to another, it is not the case for the errors on the beacons' locations.

We notice that the Gaussian mixture ($gaussMixt8$ and $gaussMixt16$) approaches fail to correctly initialize the beacons. In this dataset, the odometry has a good accuracy whereas the ranging system is more noisy. It explains why it is more challenging here to initialize accurately the beacons. Moreover, the distances measured can be up to 60 m with a greater noise for far ranges. With longer ranges, the Gaussian mixture approach has more difficulties to represent the different hypotheses on the circle because there is more space to cover. The $trilat10$ benefits again from the good odometry accuracy to obtain good results. Our approaches are still amongst the methods with the lowest errors for both the trajectory and the map. The difference between the two versions stays very close, as shown in Fig. 4 for the $compositeAug$ method. But this time, the $compositeAug$ method performs better than the $composite$ method. Thereby, although theoretically the $compositeAug$ method should give better performances, the differences with it's simplified version $composite$ are very small.

Method	Trajectory RMSE (m)	Map RMSE (m)	Execution time (s)	endInit
$trilat10$	1.44	2.89	0.71	6428
$gaussMixt8$	2.17	16.12	1.00	789
$gaussMixt16$	1.42	13.63	2.50	789
$partFilter360$	1.00	3.38	0.87	884
$composite$	1.07	2.97	0.94	813
$compositeAug$	1.03	2.87	0.87	813
$compositeAug$ w/o Mahalanobis test	1.14	2.90	0.85	813

TABLE II

COMPARISON OF DIFFERENT INITIALIZATION TECHNIQUES FOR THE PLAZA1 DATASET

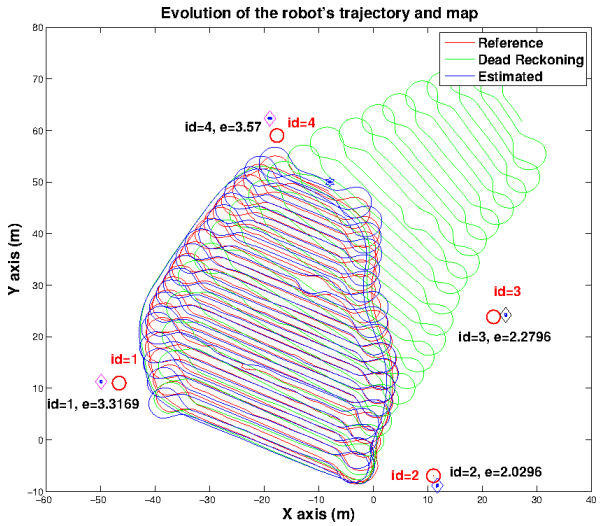

Fig. 4. Final robot's trajectory and beacons' positions for the Plaza1 dataset for the $compositeAug$ method. e corresponds to the final error of the beacon. The RMSE for the final trajectory is 1.03 m.

V. CONCLUSIONS

This paper introduced a method to initialize the beacons' positions in an EKF for the case of RO-SLAM in a planar world. The method combines a short delayed initialization with a limited (up to two) number of hypotheses for each beacon. The experiments showed that the approach correctly initializes the beacons, and accurately reconstructs the robot's trajectory and the beacons' locations. But, as for other initialization techniques, the result greatly depends on the accuracy of the range measurements used. The configuration of the robot's positions used for the initialization also influence the obtained solutions. Thus, it would be interesting to actively control the robot's trajectory in order to improve the initial estimate of the beacons. Other improvements can be made by using a better model of the range measurements, and by exploiting the inter-beacon range measurements as in [20]. It would reduce the map estimation uncertainty and indirectly improve the robot localization as written in [10].

REFERENCES

[1] R. C. Smith and P. Cheeseman, "On the representation and estimation of spatial uncertainty," *The International Journal of Robotics Research*, 1986.

[2] J. Graefenstein and M. Bouzouraa, "Robust method for outdoor localization of a mobile robot using received signal strength in low power wireless networks," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2008.

[3] Y. Zhou, "An efficient least-squares trilateration algorithm for mobile robot localization," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2009.

[4] J. J. Leonard, R. J. Rikoski, P. M. Newman, and M. Bosse, "Mapping partially observable features from multiple uncertain vantage points," *The International Journal of Robotics Research*, 2002.

[5] E. Olson, J. J. Leonard, and S. Teller, "Robust range-only beacon localization," *IEEE Journal of Oceanic Engineering*, 2006.

[6] J.-L. Blanco, J. Gonzalez, and J.-A. Fernandez-Madrigal, "A pure probabilistic approach to range-only slam," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2008.

[7] F. Caballero, L. Merino, and A. Ollero, "A general gaussian-mixture approach for range-only mapping using multiple hypotheses," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2010.

[8] F. Fabresse, F. Caballero, I. Maza, and A. Ollero, "Undelayed 3d ro-slam based on gaussian-mixture and reduced spherical parametrization," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2013.

[9] J. Djugash and S. Singh, "A robust method of localization and mapping using only range," in *Experimental Robotics*, 2009.

[10] A. Torres-Gonzalez, J.-d. Dios, and A. Ollero, "Efficient robot-sensor network distributed seif range-only slam," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2014.

[11] A. Kehagias, J. A. Djugash, and S. Singh, "Range-only slam with interpolated range data," *Technical report*, 2006.

[12] F. Herranz, A. Llamazares, E. Molinos, and M. Ocana, "A comparison of slam algorithms with range only sensors," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2014.

[13] J. Huang, D. Millman, M. Quigley, D. Stavens, S. Thrun, and A. Aggarwal, "Efficient, generalized indoor wifi graphslam," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2011.

[14] B. Boots and G. J. Gordon, "A spectral learning approach to range-only slam," in *Int. Conf. on Machine Learning (ICML)*, 2013.

[15] P. Lourenco, P. Batista, P. Oliveira, C. Silvestre, and C. P. Chen, "Sensor-based globally exponentially stable range-only simultaneous localization and mapping," *Robotics and Autonomous Systems*, 2015.

[16] J. Castellanos, R. Martinez-Cantin, J. Tardos, and J. Neira, "Robocentric map joining: Improving the consistency of ekf-slam," *Robotics and Autonomous Systems*, 2007.

[17] P. Batista, C. Silvestre, and P. Oliveira, "Single range aided navigation and source localization: Observability and filter design," *Systems & Control Letters*, 2011.

[18] J. Djugash, B. Hamner, and S. Roth, "Navigating with ranging radios: Five data sets with ground truth," *Journal of Field Robotics*, 2009.

[19] S. Cousins, "Welcome to ros topics [ros topics]," *IEEE Robotics Automation Magazine*, 2010.

[20] J. Djugash, S. Singh, G. Kantor, and W. Zhang, "Range-only slam for robots operating cooperatively with sensor networks," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2006.