Bearing-only SLAM: A fast algorithm without using Extended Kalman filter

[1]D.G.ARSENIEV, [2]N.A. BERKOVSKII
[12]Sankt-Peterburgskij Gosudarstvennyj Politehniceskij Universitet,
St. Petersburg, Polytechnicheskaya, 29
RUSSIAN FEDERATION
[1]imop@imop.spbstu.ru, [2]berkovsk@mail.ru, http:// www.spbstu-eng.ru

Abstract: - In this paper a new method for solving 2D Bearing-only SLAM is proposed. We use only Sequential Monte Carlo Methods to estimate current positions of the robot and for determining the landmarks' coordinates. The main advantages of the proposed method are the high speed and the trivial generalization to 3D case. Our method has linear complexity growth with respect to number of the landmarks.

Key-Words: Bearing-only SLAM, Sequential Monte Carlo Methods, Patricle filters, Rao-Blackwellisation method, Fast SLAM.

1 Introduction

Recently, for solving SLAM problem is widely used the so-called Fast-Slam method [1, 2]. It uses Sequential Monte Carlo Methods (SMC) to estimate the robot's current position and Extended Kalman Filter (EKF) for determining the landmarks' coordinates. A similar technique, which is called Rao-Blackwellisation method, is applied in other applications, for instance, see [4, 5]. It is known that Fast-Slam is effective in case the mobile robot is equipped with a range finder [2]. However, if bearing measurements are only available, Fast-Slam may not have a good performance. This is because the uncertainty in the disposition of the landmarks is too high while the linearization used in EKF doesn't yield big errors, in general, for small uncertainties exclusively. The main reason for the large errors produced Fast-Slam is a failed initialization, which takes place if some landmarks are disposed too close or too far from the robot. One remedy to overcome this problem is Gaussian Sum Filter (GSF) [2, 7] which uses the set of EKFs to estimate the landmark's positions. However, GSF has an exponential growth of the complexity with respect to the number of landmarks [3]. In this paper, we propose a fully SMC-based method for Bearing Only SLAM, which doesn't use linearization and, for this, produces more accurate results than Fast-Slam. At the same time, our method inherits the idea of the state space decomposition from Fast-Slam. In this regard, it is like the method proposed in [3], but we don't use the trapezoids to model the uncertainties of the landmarks. Instead we are modeling landmarks' uncertainties as a set of the separate particle filters [4]. Doing in that way, we lately have got three different methods for solving Bearing-only problem. For these methods, the higher speed of processing per one step of estimation, the larger bias of the estimates. However at the reasonable values of the noises included in the model this bias may be ignored. In this paper, we represent a quickest algorithm of all the three approaches. It is very effective if system noises are moderate. Unlike the GSF, the complexity of our method grows linearly with increasing of landmarks' number. We deal with a 2D environment but the technique developed can be generalized to 3D space without any changes.

It is worth to note, Bearing-only SLAM is attractive choice due to inexpensive equipping of the robot, since the range finder is not used but the single camera is only required [2]. There are two categories of the methods for solving Bearing-only SLAM. They are delayed and un-delayed methods. In delayed methods, the estimation of the robot's and landmarks positions is postponed until the reliable base-line is reached [3, 6]. On the contrary, an un-delayed method modifies those positions as soon as a new measurement is available. In this paper we propose an un-delayed method.

The paper is organized as follows: in section 2 the problem is formulated and denotes are introduced, in the section 3 our algorithm and the version Fast SLAM used for comparison are described. Section 4 is devoted to simulation results, and section 5 concludes the paper.

2 Problem Formulation

We use the kinematic discrete model of the robot's motion. Let a mobile robot, whose coordinates are

$\mathbf{r} = (r_x, r_y)$, moves on the $x-y$ plane along a scheduled trajectory (Fig.1).

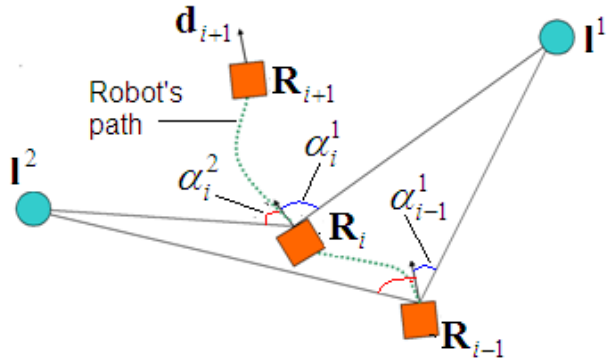


Fig.1 The motion of robot and taking of the measurements in bearing-only SLAM.

The robot starts at the known point $\mathbf{r}_0 = (r_{x0}, r_{y0})$ and then makes N_{st} consecutive movements (steps). By θ denote the orientation of the robot with respect to the *x* axis. Suppose there are N_l landmarks on the plane and let $\mathbf{l}^s = (l_x^s, l_y^s)$, $s = 1,2,\dots N_l$ be their coordinates. The robot's motion can be described as follows:

$$\Delta\mathbf{r}_i = \left((\rho_i + e_i^\rho)\cos\theta_{i-1}, (\rho_i + e_i^\rho)\sin\theta_{i-1}\right), \quad (1)$$

$$\theta_i = \theta_{i-1} + \Delta\theta_i + e_i^\theta . \quad (2)$$

where $\rho_i, \Delta\theta_i$ are control parameters forming the robot's path, $i = \overline{1, N_{st}}$. The system noise processes e_i^ρ, e_i^θ are described by their probability density functions (p.d.f.) $\mathbf{N}(e_i^\rho, \sigma_i^\rho)$ and $\mathbf{N}(e_i^\theta, \sigma_i^\theta)$, where $\mathbf{N}(x, \sigma)$ is the p.d.f. of the centered normal distribution with variance σ^2 and argument *x*. The random variables e_i^ρ and e_j^θ (for any *i*, *j*) as well as e_i^ρ and e_j^ρ, e_i^θ and e_j^θ (for $i \neq j$) are assumed to be independent of each other. Let $\mathbf{r}_i = (r_{xi}, r_{yi})$ be the position of the robot at the step *i*, then we have

$$\mathbf{r_i} = \mathbf{r_{i-1}} + \Delta\mathbf{r_i}, \quad (3)$$

where $\Delta\mathbf{r}_i$ are defined by (1) и (2). The model defined by (1) – (3) is more suitable for a walking robot, but it may be also used for a wheeled robot. The initial orientation θ_0 is known. Let α_i^s be the angle between the vectors $\mathbf{d}_i = (\cos\theta_i, \sin\theta_i)$ and $\mathbf{l}^s - \mathbf{r}_i$, $\alpha_i^s \in (-\pi; \pi]$. More precisely,

$$\alpha_i^s = \arccos\left(\frac{\mathbf{d}_i \cdot (\mathbf{l}^s - \mathbf{r}_i)}{\|\mathbf{l}^s - \mathbf{r}_i\|}\right) \operatorname{sgn}\begin{vmatrix} \cos\theta_i & \sin\theta_i \\ l_x^s - r_{xi} & l_y^s - r_{yi} \end{vmatrix}, \quad (4)$$

where $\|\ \|$ is a vector magnitude. For all $i = \overline{0, N_{st}}$ and $s = \overline{1, N_l}$ the noisy measurements m_i^s are supposed to be available and

$$m_i^s = \alpha_i^s + e_i^\alpha , \quad (5)$$

where e_i^α are independent random variables with p.d.f. $\mathbf{N}(e_i^\alpha, \sigma_i^\alpha)$.

Let us introduce the notations $\mathbf{R}_i = (\mathbf{r}_i, \theta_i)$, $\mathbf{L} = (\mathbf{l}^1, \mathbf{l}^2, \dots, \mathbf{l}^{N_r})$, $\mathbf{S}_i = (\mathbf{R}_i, \mathbf{L})$, and $\mathbf{M}_i = \{m_k^s\}$, where $s = \overline{1, N_l}$ and $k = \overline{1, i}$. In this paper, our aim is to estimate recursively in time the conditional mean $\mathrm{E}_{f(\mathbf{S}_i|\mathbf{M}_i)}\mathbf{S}_i$, $i = \overline{1, N_{st}}$, where $f(\mathbf{S}_i|\mathbf{M}_i)$ is the p.d.f. of the posterior distribution of the state vector $\mathbf{S}_i$. Hereinafter, the estimate of $\mathrm{E}_{f(\mathbf{S}_i|\mathbf{M}_i)}\mathbf{S}_i$ is denoted as $\overline{\mathbf{S}_i}$.

3 Description of the proposed method and Fast SLAM.

The following factorization (6) of the conditional p.d.f. is needed for the sequel:

$$f(\mathbf{S}_i|\mathbf{M}_i) = f(\mathbf{L}|\mathbf{R}_i, \mathbf{M}_i) f(\mathbf{R}_i|\mathbf{M}_i) \quad (6)$$

The formula (6) follows from the Bayesian rules [3,4]. Using (6), we can estimate the parts $\mathbf{R}_i$ and $\mathbf{L}$ of the state vector $\mathbf{S}_i$, in the certain sense, separately.

3.1 Description of the proposed method

3.1.1 Initialization:

For all $s \in \overline{1, N_l}$ the random vectors $\mathbf{l}_{0(j)}^s = (l_{0(j)x}^s, l_{0(j)y}^s)$, $j = \overline{1, N_l^g}$ are generated according to a prior distribution with p.d.f. $f(\mathbf{l}_0^s)$. The estimates $\overline{\mathbf{l}_0^s} = (N_l^g)^{-1} \sum_{j=1}^{N_l^g} \mathbf{l}_{0(j)}^s$, $s \in \overline{1, N_l}$ are calculated. Suppose $\mathbf{R}_{0(k)} = \mathbf{R}_0$, $k \in \overline{1, N_r}$, that is, the initial state of the robot is the same for all the robot's trajectories simulated. To sample $\mathbf{l}_{0(j)}^s = (l_{0(j)x}^s, l_{0(j)y}^s)$, at first the random variables $u_{(j)}^s$ are sampled from the uniform distribution over the predefined range $[u_{\min}, u_{\max}]$ of the landmarks' depth. Then, for all $s \in \overline{1, N_l}$ sample the random

variables $\varphi^s_{(j)}$ according to $\mathbf{N}\left(\overline{\varphi^s}, \sigma_0^\alpha\right)$, where $\overline{\varphi^s}$ is the angle between the x axis and the ray forming angle m_0^s with the initial vector $\mathbf{d}_0 = (\cos\theta_0, \sin\theta_0)$. Finally, put

$$\begin{cases} l^s_{0(j)x} = u^s_{(j)} \cos\left(\varphi^s_{(j)}\right) + r_{x0} \\ l^s_{0(j)y} = u^s_{(j)} \sin\left(\varphi^s_{(j)}\right) + r_{y0} \end{cases}, \quad j = \overline{1, N_l^g}, \quad s \in \overline{1, N_l} \quad (7)$$

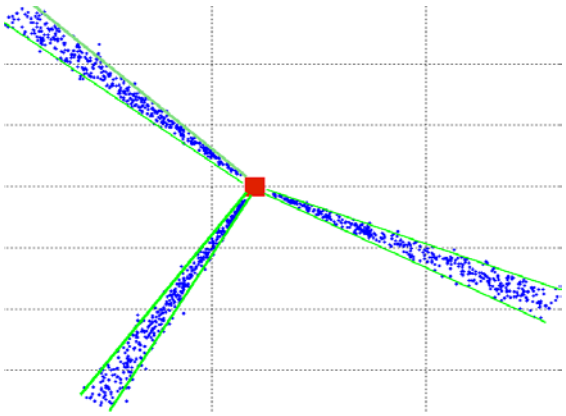


Fig.2 Initialization

Result of applying (7) is shown in Fig.2 for the case of the $N_l = 3$.

3.1.2 Updating the robot's trajectories

For all $k \in \overline{1, N_r}$ the independent Gaussian variables $e^\rho_{i(k)}$ и $e^\rho_{i(k)}$ are generated. Applying (1-3) to all of the vectors $\mathbf{R}_{i-1(k)}$, we get the set of the vectors $\mathbf{R}^*_{i(k)}, k \in \overline{1, N_r}$ whose p.d.f. is approximately $f\left(\mathbf{R}_i \middle| \mathbf{M}_{i-1}\right)$. Calculate the angles $\alpha^s_{i(k)}$ between $\mathbf{d}^*_{i(k)}$ and $\overline{\mathbf{l}^s_{i-1}} - \mathbf{r}^*_{i(k)}$ (Fig.3).

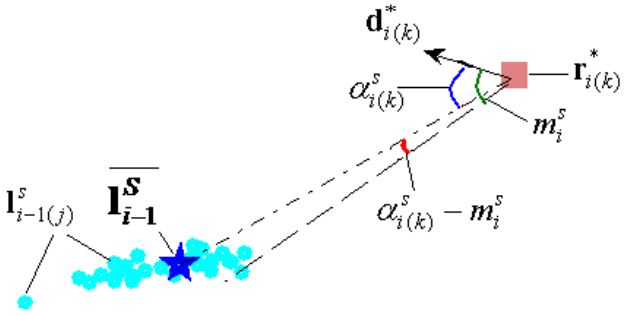


Fig.3 Updating of robot's trajectories

Let $q_{(k)} = Q_{(k)} \left(\sum_{k=1}^{N_r} Q_{(k)} \right)^{-1}$ are weights, where

$$Q_{(k)} = \prod_{s=1}^{N_l} \mathbf{N}\left(\left(\alpha^s_{i(k)} - m^s_i\right)_{(-\pi,\,\pi]},\ h\sigma^\alpha_i \right) \quad (8)$$

In (8) $h > 1$ is design parameter. The subscript at $\left(m^s_{i(k)} - m^s_i\right)$ means we choose the value of angles falling in $(-\pi, \pi]$.

Now, the random vectors $\mathbf{R}_{i(k)}$ are modeled according to the p.d.f.

$$\sum_{m=1}^{N_r} q_{(m)} \delta\left(\mathbf{R}_{i(k)} - \mathbf{R}^*_{i(m)}\right), \quad (9)$$

where $\delta(.)$ is the Dirac delta function. The uniform discrete distribution based on the set $\left\{\mathbf{R}_{i(k)}\right\}_{k=1}^{N_r}$, represents the desired posterior distribution with p.d.f. $f\left(\mathbf{R}_i \middle| \mathbf{M}_i\right)$. At last, we obtain the estimate $\overline{\mathbf{R}_i} = \sum_{k=1}^{N_r} q_{(k)} \mathbf{R}^*_{i(k)}$.

3.1.3 Updating the landmarks' positions:

For all $s \in \overline{1, N_l}$ and $j = \overline{1, N_l^g}$ calculate the angles $\alpha^s_{i(j)}$ between the vectors $\overline{\mathbf{d}}_i = (\cos\overline{\theta}_i, \sin\overline{\theta}_i)$ and $\mathbf{l}^s_{i-1(j)} - \overline{\mathbf{r}_i}$, where $\overline{\mathbf{r}_i}$ and $\overline{\theta}_i$ are components of the compound vector $\overline{\mathbf{R}_i}$ (Fig.4).

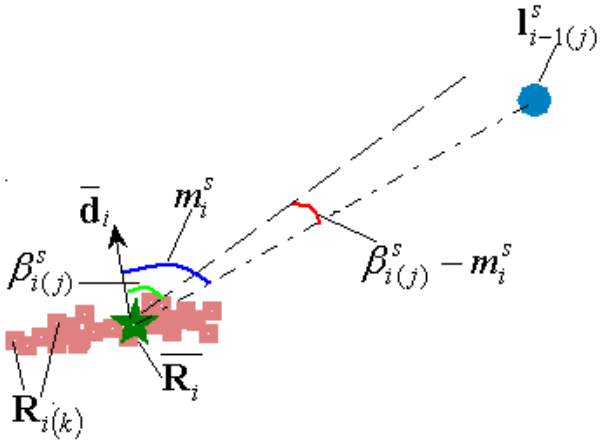


Fig.4 Updating of landmarks

Then, count the weights $w^s_{(j)} = W_{(j)} \left(\sum_{j=1}^{N_l^g} W_{(j)} \right)^{-1}$, where

$$W_{(j)} = \mathbf{N}\left(\left(\alpha^s_{i(j)} - m^s_i\right)_{(-\pi,\,\pi]},\ \sigma^\alpha_i \right) \quad (10)$$

Now, the vectors $\mathbf{l}^s_{i(j)}$ are simulated according to the p.d.f.

$$\sum_{m=1}^{N_l^g} w_{(m)}^s \delta\left(\mathbf{l}_{i(j)}^s - \mathbf{l}_{i-1(m)}^s\right) \quad (11)$$

The uniform discrete distribution based on the set $\left\{\mathbf{l}_{i(j)}^s\right\}_{j=1}^{N_l^g}$, $s \in \overline{1, N_l}$ represents the desired posterior distribution, whose p.d.f. is $f\left(\mathbf{L}\middle|\mathbf{R}_i, \mathbf{M}_i\right)$. Finally, calculate the estimates $\overline{\mathbf{l}_i^s} = \sum_{j=1}^{N_l^g} w_{(j)}^s \mathbf{l}_{i(j)}^s$ and $\overline{\mathbf{L}_i} = \left(\overline{\mathbf{l}_i^1}, \overline{\mathbf{l}_i^2}, \ldots, \overline{\mathbf{l}_i^{N_l}}\right)$. Finally, the i[th] estimate of the state vector is $\overline{\mathbf{S}_i} = \left(\overline{\mathbf{R}_i}, \overline{\mathbf{L}_i}\right)$.

3.1.4 Complexity analysis

It can be seen from subsections 3.1.2 and 3.1.3 that we have to evaluate the normal distribution $N_l N_r$ times in (8) and $N_l N_l^g$ times in (12), total is $N_l\left(N_r + N_l^g\right)$. If we considered all of the interrelations between robot's and landmarks' particles we would have a much larger number $N_l\left(N_r N_l^g\right)$ and processing would be too slow. Using only averages $\overline{\mathbf{l}_{i-1}^s}$ in (8) and $\overline{\mathbf{r}_i}$ in (11) for evaluating $\alpha_{i(k)}^s$ and $\alpha_{i(j)}^s$, we increase speed of computation. This approximation produces some additional bias of estimate, of course. The design parameter h in (8) is needed to decrease that bias. In this paper we use the value $h = 3$, at which most precise results were obtained. If σ_i^ρ and σ_i^θ are relatively small, the bias may be ignored. For large values of σ_i^ρ and σ_i^θ it is better to use methods based on computing all of the $N_l\left(N_r N_l^g\right)$ angles between $\mathbf{d}_{i(k)}^*$ and $\mathbf{l}_{i-1(j)}^s - \mathbf{r}_{i(k)}^*$. Then the number of measurements should be decreased for the speed. Further, it is clear that complexities of the resampling processes in (9) and (11) depend on N_l linearly, therefore we have linear growth of complexity our algorithm with respect to the number of landmarks. This fact is also confirmed by simulation.

3.2 Description of Fast SLAM

Below, the particular case of the Fast Slam 1.0 algorithm is described. The initialization and updating for robot's pathes are depicted as they are carried in our numerical examples, updating of the landmarks follow [1, 2]

3.2.1 Initialization: The random vectors $\mathbf{l}_{0(k)}^s = \left(l_{0(k)x}^s, l_{0(k)y}^s\right)$ ($k = \overline{1, N}$; $s \in \overline{1, N_l}$) are generated according to $\mathrm{N}\left(\mathbf{l}_0^s, \overline{\mathbf{l}_0^s}, \mathbf{\Sigma}_0^s\right)$, where $\mathrm{N}\left(\mathbf{x}, \bar{\mathbf{x}}, \mathbf{\Sigma}_\mathbf{x}\right)$ is the p.d.f. of Bivariate Normal Distribution with mean $\bar{\mathbf{x}}$, covariance matrix $\mathbf{\Sigma}_\mathbf{x}$, and argument $\mathbf{x}$. For all $s \in \overline{1, N_l}$ the values of $\overline{\mathbf{l}_0^s}$ and $\mathbf{\Sigma}_0^s$ coincide with the sample mean and the sample covariance matrix of the set $\mathbf{l}_{0(k)}^s = \left(l_{0(k)x}^s, l_{0(k)y}^s\right)$, which is defined by (7). Put $\mathbf{R}_{0(k)} = \mathbf{R}_0$, $k = \overline{1, N}$. Here, N is number of "particles" [4].

3.2.2 Updating

Updating of the robot's pathes is the same as in the proposed method (subsection 3.1.2) except for $\alpha_{i(k)}^s$ are the angles between $\mathbf{d}_{i(k)}^*$ and $\mathbf{l}_{i-1(k)}^s - \mathbf{r}_{i(k)}^*$, and $k \in \overline{1, N}$ all over. Suppose the parameters $\overline{\mathbf{l}_{i-1(k)}^s}$ and $\mathbf{\Sigma}_{i-1(k)}^s$ ($k = \overline{1, N}$; $s \in \overline{1, N_l}$) have been already known. If the robot's state is $\mathbf{R}_{i(k)}$, we have $\mathbf{l}_{i(k)}^s = \mathbf{l}_{i-1(k)}^s$; $m_{i(k)}^s = \alpha_i^s\left(\mathbf{l}_{i(k)}^s\right) + e_{i(k)}^\alpha$ and

$$\alpha_i^s\left(\mathbf{l}_{i(k)}^s\right) = \arccos\left(\frac{\mathbf{d}_{i(k)} \cdot \left(\mathbf{l}_{i(k)}^s - \mathbf{r}_{i(k)}\right)}{\left\|\mathbf{l}_{i(k)}^s - \mathbf{r}_{i(k)}\right\|}\right) \cdot \operatorname{sgn}\begin{vmatrix} \cos\theta_{i(k)} & \sin\theta_{i(k)} \\ l_{i(k)x}^s - r_{i(k)x} & l_{i(k)y}^s - r_{i(k)y} \end{vmatrix} \quad (12)$$

For any s, consider $\alpha_i^s\left(\mathbf{l}_{i(k)}^s\right)$ from (12) as function of only $\mathbf{l}_{i(k)}^s$. Then, for all $k \in \overline{1, N}$ apply EKF with linearization $\alpha_i^s\left(\mathbf{l}_{i(k)}^s\right)$ at the predicted point $\mathbf{l}_{i-1(k)}^s$. We have standard equations for EKF:

$$\mathbf{l}_{i(k)}^s = \mathbf{l}_{i-1(k)}^s + \mathbf{K}_{(k)}\left(\alpha_i^s\left(\mathbf{l}_{i(k)}^s\right) - m_i^s\right)_{(-\pi, \pi]} \quad (13)$$

$$\mathbf{\Sigma}_{i(k)}^s = \mathbf{\Sigma}_{i-1(k)}^s - \mathbf{K}_{(k)}\mathbf{G}_{(k)}\left(\mathbf{\Sigma}_{i-1(k)}^s\right)^t \quad (14)$$

$$\mathbf{K}_{(k)} = \mathbf{\Sigma}_{i-1(k)}^s \mathbf{G}_{(k)}^t \left(\mathbf{G}_{(k)}\mathbf{\Sigma}_{i-1(k)}^s \mathbf{G}_{(k)}^t + \left(\sigma_i^\alpha\right)^2\right)^{-1} \quad (15)$$

where $\mathbf{G}_{(k)}$ is the gradient of the function $\alpha_i^s\left(\mathbf{l}_{i(k)}^s\right)$ computed at $\mathbf{l}_{i-1(k)}^s$. Note, $\mathbf{G}_{(k)}$ is a 2×1 matrix. The uniform discrete distribution based on the set of the compound vectors $\left\{\left(\mathbf{R}_{i(k)}, \mathbf{l}_{i(k)}^s\right)\right\}_{k=1}^N$, $s \in \overline{1, N_l}$ represents the desired joint distribution with p.d.f. $f\left(\mathbf{S}_i\middle|\mathbf{M}_i\right)$. By definition, put

$\overline{\mathbf{R}_i} = \sum_{k=1}^{N} \mathbf{R}_{i(k)}$, $\overline{\mathbf{l}_i^s} = \sum_{k=1}^{N} \mathbf{l}_{i(k)}^s$, $\overline{\mathbf{L}_i} = \left(\overline{\mathbf{l}_i^1}, \overline{\mathbf{l}_i^2}, \ldots, \overline{\mathbf{l}_i^{N_l}}\right)$.
Finally the i[th] estimate of the state vector $\overline{\mathbf{S}_i} = \left(\overline{\mathbf{R}_i}, \overline{\mathbf{L}_i}\right)$.

4 Simulation

We consider the discrete model of circular motion. The robot carries 36 steps trying to stay on the unit circle during its walking. The noises make the robot change its path a bit. It starts at (1,0) and plans return to the same point. For it, the length of its steps $\rho_i = 2\pi/36 = 0{,}174$ and angles $\Delta\theta_i = 10^\circ$ (see Section 2). In our experiment all of the linear sizes are expressed in relative numbers, so if other step length is needed one can just multiply by appropriate factor all of the model linear parameters. The values of the parameters are $N_{st} = 36$; $N_l = 6$; $N_r = N = 500$; $N_l^g = 800$; $\sigma_i^\alpha = 1^\circ$; $\sigma_i^\rho = 0.03\rho_i = 0.005$; $\sigma_i^\theta = 0.3^\circ$; $u_{\min} = 0.5$; $u_{\max} = 6$. The landmarks are sampled from the uniform distribution so that their distances from the robot are within $[u_{\min}, u_{\max}]$ for all $i = \overline{1, N_{st}}$. We randomly choose the 3 landmarks into the unit circle and 3 landmarks out of one. Modeling has shown the position of a landmark with respect to the circle affects the estimation accuracy. The typical results after running the proposed method and Fast Slam are shown in Fig. 5.

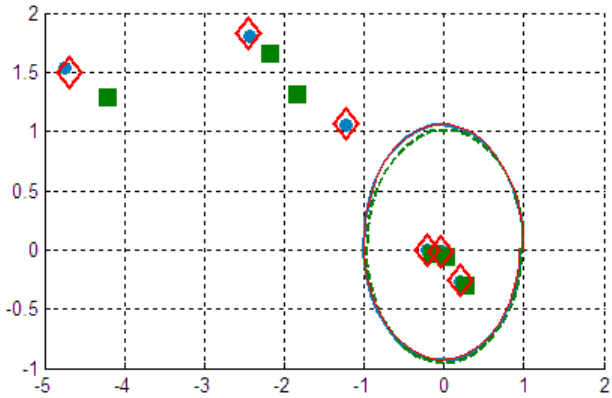


Fig.5 The consecutive estimates of the robot's positions and the localization of the landmarks estimated at last step. red− the proposed method; green− Fast Slam, blue− the true values.

Fig.5 shows that the proposed method works more accurately than Fast SLAM. Moreover, the worse situation for Fast SLAM, when a landmark is close to the robot is depicted in Fig.6. In Fig.6, one can see that the error produced by Fast Slam is extremely large, while the proposed method estimates the landmark's position very precisely. The main reason is failed initialization which is shown in Fig.7. Since the proposed method has stochastic nature, we 2000 times randomly generated sets of the 6 landmarks (3 into and 3 out of the circle). The results are represented as mean and median of absolute error and shown in Table 1.

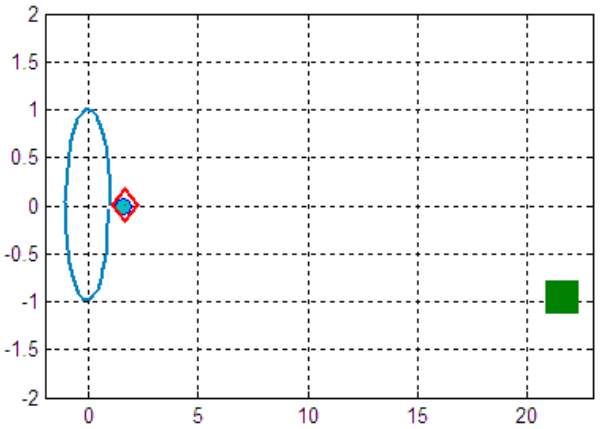


Fig.6 The estimated position of the landmark located at (1.6, 0): red − the proposed method; green −Fast Slam, blue− the true values.

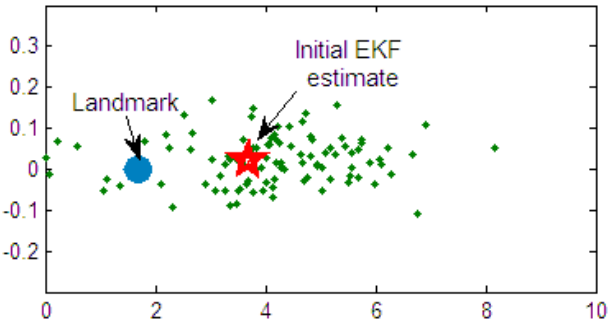


Fig.7 Failed initialization for Fast SLAM

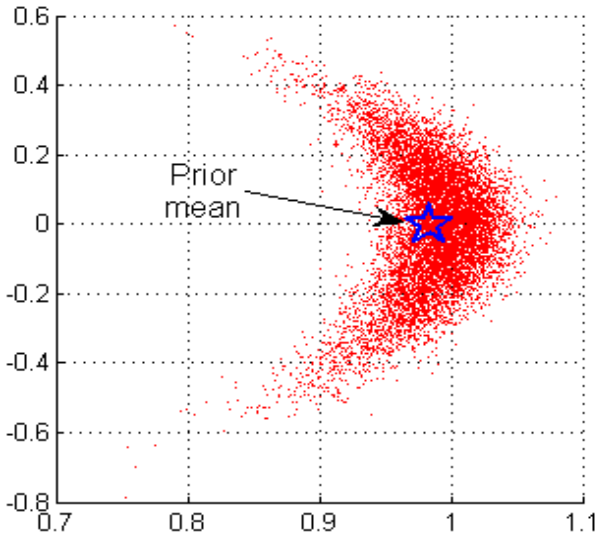
Table 1. Results of 2000 running of the proposed method

Estimator of abs error/	Landmarks into the circle	Landmarks out of the circle	The robot's position
Mean	0.022	0.12	0.025
Median	0.019	0.085	0.022

As it follows from Table 1, for both type of the landmarks the relative errors are not bigger then 3% typical linear size of the corresponding areas. To estimate the accuracy of the robot's coordinates computation, consider the prior distribution of the final points of the robot. We have modeled 100000 trajectories (without taking of measurements) at the

noise values $\sigma_i^{\rho} = 0.03\rho_i = 0.005$; $\sigma_i^{\theta} = 0.3^{\circ}$ and the set of the final points and the prior mean is shown in Fig.8. The mean distance these points from the prior mean is 0.15, therefore the proposed method decreases the prior uncertainty more than 7 times. We used Matlab and the run time was approximately 0.06 s. per one step in the case of 6 landmarks. (The CPU clock speed was 3GHz). Apparently, this value can be improved by program optimization.

Additionally, we carried set of computational experiments at the different values of the noises σ_i^{α}; σ_i^{ρ}; σ_i^{θ}. These experiments have shown that our method estimates reliably standard deviation for the current final robot position, but doesn't yield a good estimate for the current final landmarks standard deviations. In our next work, we are going to represent the SMC-based fast method which will estimate properly landmarks' uncertainties.



5 Conclusion

We have proposed the fast method for solving Bearing-only SLAM, which has linear complexity growth with respect to number of the landmarks. The main advantages of the proposed method are the high speed and the trivial generalization to 3D case, which can be seen from the algorithm structure. In the future, we intend to modify this method in order to achieve a reasonable accuracy in estimating of the current uncertainties landmarks' position.

References:

[1] S. Thrun, M. Montemerlo, D. Koller, B.Wegbreit, J. Nieto, and E. Nebot. Fastslam: An efficient solution to the simultaneous localization and mapping problem with unknown data association, 2004. // http://robots.stanford.edu/papers/Thrun03g.html

[2] K.E.Bekris, M.Glick, L.E. Kavraki. Evaluation of Algorithms for bearing-only SLAM// ICRA 2006: pp.1937-1943

[3] M. H. Mirabdollah, B. Mertching. Bearing only SLAM: a new particle filter based approach// Proceedings of the Third international conference on Autonomous and Intelligent Systems, 2012, pp.116 -125

[4] Doucet A. Sequentiual Monte-Carlo methods in practice. /Doucet.A., Freitas N., Gordon N.− N.Y., Shpringer-Verlag, 2001.

[5] Stepanov O.A., Toropov A.B. Application of the Monte Carlo Methods with Partial Analytical Integration Techniques to the Problem of Navigation System Aiding. Proceeding of 20-th Saint Petersburg International Conference on Integrated Navigation Systems. Russia 2013, pp.299-301

[6] M Bryson, S Sukkarieh Bearing-only SLAM for an airborne vehicle- Australasian Conf. Robot. Autom.(ACRA 2005), Sydney , 2005

[7] Kwok, N.M. Bearing Only SLAM using a SPRT Based Gaussian Sum Filter.//ICRA 2005, pp. 1109 – 1114