Avrupa Bilim ve Teknoloji Dergisi
Sayı 20, S. 343-350, Aralık 2020

Araştırma Makalesi

www.ejosat.com ISSN:2148-2683

European Journal of Science and Technology
No. 20, pp. 343-350, December 2020

Research Article

Localization and Initialization Algorithms based on UWB, LiDAR and Odometry for Robotic Applications with ROS Ecosystem

Pınar Oğuz Ekim*

Izmir University of Economics, Faculty of Engineering, Departmant of Electrical and Electronics Engineering, İzmir, Turkey, (ORCID: 0000-0003-1860-4526), pinar.ekim@ieu.edu.tr

(First received 1 Haziran 2020 and in final form 15 Ekim 2020)

(**DOI:** 10.31590/ejosat.746214)

ATIF/REFERENCE: Oğuz Ekim, P. (2020). Localization and Initialization Algorithms based on UWB, LiDAR and Odometry for Robotic Applications with ROS Ecosystem. *European Journal of Science and Technology*, (20), 343-350.

Abstract

This paper describes the initialization problem along with the localization problem over the Turtlebot3 and many more mobile robots. The least squares techniques and the squared range measurements obtained from ultra-wide band (UWB) sensors are used for calculating the initial robot position. Then by exploiting the initial position, Light Detection and Ranging (LiDAR) scans and scan matching technique have been proposed to find the initial heading. Thus, the autonomous pose initialization, which is an important problem in robotic applications, is solved. The Extended Kalman Filter, which fuses UWB range measurements, odometry and Adaptive Monte Carlo Localization (AMCL) pose information, is adopted to localize the robot during its trajectory. New modules have been implemented for Robot Operating Systems (ROS) for real and simulation environments and they are made to be open source to enable wide-spread adoption. The simulation results have shown that the proposed method's Root Mean Square Error (RMSE) is 3 cm and it's almost twice better in accuracy than the benchmarked method.

Keywords: Extended kalman filter, Autonomous mobile robots, Robot navigation, Robot localization, Ultra-wide band, Lidar.

ROS Ekosistemi ile Robotik Uygulamalar için UWB, LiDAR ve Odometriye Dayalı Konumlandırma ve İlklendirme Algoritmaları

Öz

Bu çalışmada, Turtlebot3 ve daha birçok mobil robot üzerindeki konum bulma sorunu ile birlikte ilklendirme sorunu açıklanmaktadır. Ultra geniş bant (UWB) sensörlerinden elde edilen uzaklıkların kareleri ölçümleri ve en küçük kareler tekniği ilk robot konumunu hesaplamak için kullanılır. Daha sonra bu başlangıç pozisyonunundan yararlanarak, ilk yönelim açısını bulmak için Işık Algılama ve Uzaklık (LiDAR) sensörünün taramaları ve tarama eşleştirme tekniği önerilmiştir. Böylece, robotik uygulamalarda önemli bir sorun olan başlangıçtaki otonom konumlandırma ve yönelim açısını bulma çözüldü. UWB uzaklık ölçümleri, kilometre odometre ve Uyarlanabilir Monte Carlo Lokalizasyon (AMCL) poz bilgisini birleştiren Genişletilmiş Kalman Filtresi, robotun yörüngesi sırasında konumu bulmak için benimsenmiştir. Gerçek ve simülasyon ortamları için Robot İşletim Sistemleri (ROS) için yeni modüller uygulanmıştır ve geniş çapta benimsenmesini sağlamak için açık kaynak olarak yapılmıştır. Simülasyon sonuçları, önerilen yöntemin Kök Ortalama Kare Hatasının (RMSE) 3 cm olduğunu ve kıyaslama yöntemden neredeyse iki kat daha iyi olduğunu göstermiştir.

Anahtar Kelimeler: Genişletilmiş kalman filtresi, Otonom mobil robotlar, Robot navigasyonu, Robot konumlandırması, Ultra geniş bant, Lidar.

* Corresponding Author: pinar.ekim@ieu.edu.tr

1. Introduction

Autonomous vehicles and robots are trending topics. Main reason for that might be some advantages that these systems introduce to the world. Some of the advantages are decreasing human influence in the industry while keeping people away from dangerous jobs such as industrial welding or disaster response and increasing production rates. However, in order to expand the work fields of robots, well designed autonomous robotic systems are required. There is one important question that needs to be answered to be able to achieve this, "Where am I?". Given a map of the environment and sensors data the problem of determining the pose of a robot is localization (Zhang et al., 2009 and Fox et al., 1999). For example, a recent work for an optimal robot path planning in the field of robotics and automation proposes Particle Swarm Optimization (PSO) which needs the current location of the robot to plan the optimal path (Beşkirli and Tefek, 2019).

Over the years different techniques and algorithms are developed to solve the localization problem accurately like Kalman filter (Jetto et al., 1999), particle filter (Vlassis et al., 2002), Monte Carlo Localization (Dellaert et. al, 1999). Each method has pros and cons. The Kalman Filter tends to be less accurate compared to the particle filter whereas the particle filter may require more computation power depending on implementation since it uses multiple particles to represent robot's possible location and move them in time according to the information coming from sensors.

A robust solution is required to increase the autonomy and adaptability of mobile robots in different circumstances and expand the range of applications (Payá et al., 2017). To address the mentioned problems, some relevant information about the environment that the robot moves is necessary. This information is a priori unknown. Therefore, to extract the necessary information from the environment autonomous mobile robots can be equipped with different sensorial systems (Paya et al., 2017). On A camera, GPS and odometry are utilized in the Kalman and particle filters to localise an outdoor robot (Lee et. al, 2009). A recent work which uses LiDAR sensor for point cloud-based 3D mapping uses GPS to localize the robot (Açıkel and Gökçen, 2019). Inertial Navigation System and a laser range sensor can be also exploited for localization (Luo et. al, 2014). Recently, Ultra-Wide-Band (UWB) has been mainly used for communication is considered as a promising solution for vehicle positioning (González et.al, 2009). Capability of data transmission range accurate estimation and material penetration make it suitable for indoor robotic applications.

Robot Operating System (ROS) is an open source meta-operating system that provides a message passing structure between different processes across a network (Robot Operating System, 2019). ROS is an ecosystem that allows to introduce a new sensor (or a node) easily and to have simultaneous data from different sensors. Therefore, it leads to easy implementation of different sensor fusion techniques and combining various hardware components. Furthermore, ROS has a simulator like Gazebo which is a powerful tool in the robotics field (Yılmaz and Bayındır, 2019). During the simulations and real tests, TurtleBot3 which is a ROS standard platform robot is exploited. Turtlebot3 uses odometry and Light Detection and Ranging (LiDAR) data to localize itself, map the environment and navigate.

Fusing the information, which come from different sensors, in a reliable and an efficient way is the solution of localization problem or many more robotic problems. Because when a sensor can not have measurements under specific environmental conditions, the other one can be used to decrease the error propagation. In addition of this, the error characteristics of a sensor can be fixed by exploting another sensor with different characteristics. Although, the importance of the sensor fusion is widely known, the newly developed sensor technologies and the compatible algorithms have not been studied extensively yet.

In order to achieve fully autonomous navigation, the initialization problem needs to be solved. The standard solution for this problem is to give the initial pose (position and orientation) of the robot manually. Instead, in this study, UWB sensors are proposed to be used to determine the initial position for navigation algorithms and also for indoor localization applications. UWB range measurements are obtained before robot moves and the algorithm estimates the location of the robot. Then this information is combined with LiDAR scans to calculate the angle of the robot facing relative to the map. With this accurate initial pose information, a better navigation can be possible.

In this paper, the Extended Kalman Filter (EKF) has been implemented and it fuses odometry, the pose output of the adaptive Monte Carlo localization (AMCL) algorithm of Turtlebot3 and UWB range measurements to localize the robot during its trajectory. The proposed algorithm has been tested in different trajectories where the UWB tag on the robot is within the line of sight and the non-line of sight of the UWB anchors which are distributed in an indoor environment in the simulation environment and the real environment. The tests have already brought several promising results which can be found in the next sections. Once more the importance of sensor fusion is proved. Because

The contributions are the integration of UWB nodes to the Turtlebot3 in the ROS ecosystem, to make use of these sensor measurements along with the wheel odometry and *amcl* algorithm's output in the EKF for localization, and the initialization of the LiDAR based navigation stack using UWB ranging and LiDAR scan matching. Moreover, a simulation environment is implemented with the help of the Turtlebot3 library in ROS. Appropriate GUIs are developed for creating synthetic data and testing any algorithm and visualizing the performances. They are open to public access in our GitHub websites (Bostanci et al., 2019).

2. Material and Method

2.1. Problem Formulation of the Localization Algorithm

2.1.1 The Extended Kalman Filter

The probabilistic theory has many applications in different scientific fields. However, it has a special case in robotics for interpreting and modelling the uncertainties about locating the robots, mapping and information from sensors (Dudek et. al, 2010). Within this theory, Bayesian Filters aim to estimate the states or the beliefs about the states of an environment iteratively.

A special type of Bayesian Filter, EKF, is proposed to solve the localization for robotic applications. Let x_t be the state to be estimated with time index *t*. The state is tried to be estimated by using the control inputs $u_{1:t} = \{u_t, u_{t-1}, \cdots u_1\}$ and the information or measurements, $z_{1:t} = \{z_t, z_{t-1}, \cdots z_1\}$, that come from sensors. The measurements have a partial knowledge about the state, or they might contain error. Additionally, there might be errors due to the state modelling and control inputs. The evolution of the states and the error characteristics of the measurements obey the probabilistic laws.

In robotic applications the motion model and the observation model are defined as follow:

$$x_t = g(x_{t-1}, u_t, v_{t-1})$$
$$z_t = h(x_t, w_t). \quad (1)$$

The motion noise and the observation noise are expressed with v_{t-1} and w_t, respectively. If the motion and the observation models are linear and the noises are additive, independent and identically distributed (i.i.d.) Gaussian distributions then the Kalman Filter (KF) is the optimum filter (Bar-Shalom et. al, 2004). In other words,

If the initial belief *bel(x_0)* has a Gaussian distribution:

$$x_o \sim \mathcal{N}(\mu_o, \Sigma_o)$$
$$bel(x_0) = p(x_0) = \det(2\pi\Sigma_o)^{-\frac{1}{2}} \exp\{-\frac{1}{2}(x_0 - \mu_0)^T \Sigma_o{}^{-1}(x_0 - \mu_0)\}$$

If the motion probability model is linear and the motion noise is additive i.i.d. Gaussian distribution:

$$x_t = A_t x_{t-1} + B_t u_t + v_{t-1}, v_t \sim \mathcal{N}(0, R_t)$$
$$p(x_t|x_{t-1}, u_t) = \det(2\pi R_t)^{-\frac{1}{2}} \exp\{-\tfrac{1}{2}(x_t - A_t x_{t-1} - B_t u_t)^T R_t{}^{-1}(x_t - A_t x_{t-1} - B_t u_t)\} \quad (2)$$

If the observation probability model is linear and the observation noise is additive i.i.d. Gaussian distribution:

$$z_t = C_t x_t + w_t, w_t \sim \mathcal{N}(0, Q_t)$$
$$p(z_t|x_t) = \det(2\pi Q_t)^{-\frac{1}{2}} \exp\{-\frac{1}{2}(z_t - C_t x_t)^T Q_t{}^{-1}(z_t - C_t x_t)\} \quad (3)$$

then the posterior distribution, *bel(x_t)*, is always Gaussian and the KF expresses it with the mean μ_t and the covariance Σ_t parameters at time *t*.

The estimated mean and covariance are updated using the motion model and this step is called the prediction step:

$$\bar{\mu}_t = A_t \mu_{t-1} + B_t u_t$$
$$\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t. \quad (4)$$

The predicted mean and covariance are updated using the observation model and this step is called the update step:

$$K_t = \bar{\Sigma}_t C_t (C_t \bar{\Sigma}_t C_t{}^T + Q_t)^{-1}$$
$$\mu_t = \bar{\mu}_t + K_t(z_t - C_t \bar{\mu}_t)$$
$$\Sigma_t = (I - K_t C_t)\bar{\Sigma}_{.t}. \quad (5)$$

The Kalman gain K_t in Eq. (5) shows at what proportion the measurement is used at the state update.

When the motion and the observation model are not linear, the approximate form of KF, the EKF, is used. The EKF exploits the first-degree Taylor approximations of the model functions as the local linear forms at the prediction and the update steps.

2.1.2. The Motion Model: Odometry Information

Technically, the odometry is a sensor, but it provides control inputs for the robotic applications (Dobrev et. al, 2018). The odometry motion model comprises the probability of transition between the states in the prediction step. Let the robot move δ_{trans} and rotate δ_{rot} in (*t-1*, *t*] so that $u_t = (\delta_{\text{trans}}, \delta_{\text{rot}})$. Let $x_{t-1}=(x,y,\theta)$ and $x_t=(x',y',\theta')$ be the previous and the current robot poses which have the following relationship:

$$\begin{pmatrix} x' \\ y' \\ \theta' \end{pmatrix} = \begin{pmatrix} x \\ y \\ \theta \end{pmatrix} + \begin{pmatrix} \delta_{trans}\cos\theta \\ \delta_{trans}\sin\theta \\ \delta_{rot} \end{pmatrix}. \quad (6)$$

To be employed in Eq. (4), the first derivatives of the motion model function g in Eq. (1) with respect to *u* and *x* are obtained as follows:

$$A_t = \frac{\partial g(.)}{\partial x} = \begin{bmatrix} 1 & 0 & -\delta_{trans}\sin\theta \\ 0 & 1 & \delta_{trans}\cos\theta \\ 0 & 0 & 1 \end{bmatrix}, \quad B_t = \frac{\partial g(.)}{\partial u} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix}. \quad (7)$$

2.1.3 The Observation Model: UWB Sensors

UWB sensors can measure the distance between the transmitter and the receiver from two-way time of arrival of radio signals. The indoor localization problem is very hard due to the multipath and the non-line of sight conditions. However, UWB technology provides more accurate range measurements as the direct path of the signal is easily differentiated from the multipath thanks to the ultra-wideband (Bregar et. al, 2018).

Let p_t^i and l^j be the locations of the receiver on the robot and the j^{th} transmitter in a 2D indoor environment (the extension to 3D is straightforward) respectively, then the true range between them is

$$r_t^{UWB} = \|p_t^i - l^j\| = \sqrt{(x_t^i - l_x^j)^2 + (y_t^i - l_x^j)^2}. \quad (8)$$

The environmental factors can be modelled as an independent Gaussian noise $w_t \sim N(0,\sigma_w^2)$ and added to the measurements which have the following expression:

$$z_t = r_t^{UWB} + w_t. \quad (9)$$

The first derivative of the observation model function h in Eq. (1) with respect to x is obtained for each range measurements as follows:

$$C_{j,t} = \frac{\partial h(.)}{\partial x} = \begin{bmatrix} \frac{x_t^i - l_x^j}{\sqrt{(x_t^i - l_x^j)^2 + (y_t^i - l_x^j)^2}} & \frac{y_t^i - l_y^j}{\sqrt{(x_t^i - l_x^j)^2 + (y_t^i - l_x^j)^2}} & 0 \end{bmatrix}. \quad (10)$$

2.1.4 The Observation Model: AMCL pose information

Turtlebot3 employs the standard algorithms *amcl*, *gmapping* and *move base* for the solution of localization, mapping and navigation problems respectively. The *amcl* algorithm is a special form of particle filter which uses the odometry and LiDAR information to calculate the pose of the robot (Dellaert et. al, 1999). The proposed localization method in this paper exploits the pose of *amcl* algorithm at the update step of the EKF. The employed *amcl* output has the following form

$$p_t^{i,amcl} = p_t^i + \eta_t \qquad (11)$$

where η_t is the noise of the *amcl* estimate.

2.2. Problem Formulation of the Initialization Algorithm

2.2.1 The Source Localization

Let $\boldsymbol{x} \in R^n$ be the unknown robot position, $\boldsymbol{a}_i \in R^n, i = 1, \cdots, m$ be known sensor positions (anchors), and $r_i = \|\boldsymbol{x} - \boldsymbol{a}_i\| + w_i$ be the measured distance between the source and the *i*-th anchor, where w_i denotes a noise term. Under i.i.d. noise maximizing the likelihood of observations for the source localization problem is equivalent to

$$minimize \quad \sum_{i=1}^{m} \left| \|\boldsymbol{x} - \boldsymbol{a}_i\|^p - r_i^p \right|^q. \qquad (12)$$

The case (p = 2, q = 2) is the main interest of this paper and corresponds to the cost function used in the SR-LS algorithm of (Beck et. al, 2008). The cost function for SR-LS is not a likelihood function. It is chosen because it is easy to implement, and its computational time is very suitable for practical robotic applications. Specifically, the problem solved in (Beck et. al, 2008) is

$$minimize \quad \sum_{i=1}^{m} \left| \|\boldsymbol{x} - \boldsymbol{a}_i\|^2 - r_i^2 \right|^2. \qquad (13)$$

The exact solution of Eq. (7) can be derived by writing the equivalent form as follows

$$\begin{aligned} &minimize && \|\mathbf{Ay} - \mathbf{b}\|^2 \\ &subject\ to && \mathrm{y^T Hy} + 2\mathrm{c^T y} = 0, \end{aligned} \qquad (14)$$

where $\mathrm{A} = \begin{bmatrix} -2a_1^T & 1 \\ \vdots & \vdots \\ -2a_m^T & 1 \end{bmatrix}$, $\mathrm{b} = \begin{bmatrix} r_1^2 - \|a_1\|^2 \\ \vdots \\ r_m^2 - \|a_m\|^2 \end{bmatrix}$, $H = \begin{bmatrix} I_n & 0_n \\ 0_1 & 0 \end{bmatrix}$, and $c = \begin{bmatrix} 0_n \\ -0.5 \end{bmatrix}$.

Next, compute α^* as the root of

$$\hat{y}(\alpha)^T H \hat{y}(\alpha) + 2c^T \hat{y}(\alpha) = 0, \quad \text{with } \alpha \in \left(-\frac{1}{\alpha_1}, \infty\right),$$

where α_1 is the maximum of the generalized eigenvalues (Golub et. al, 1996) of (H, $\mathrm{A^TA}$) and $\hat{y}(\alpha) = (\alpha \mathrm{H} + \mathrm{A}^T\mathrm{A})^{-1}(\mathrm{A^T b} - \alpha \mathrm{c})$. Finally, the minimizer of the objective in (7) is given by the *n* first component of $\hat{y}(\alpha^*) \in R^{n+1}$.

2.2.2 The LiDAR based heading

The navigation stack of ROS is exploited to navigate a robot in a predefined map. The navigation stack resorts to *amcl* algorithm to estimate the current pose of the robot and optimize the path to the goal pose. However, *amcl* algorithm needs a good initialization and the common way is to define an initial starting pose and to determine the goal point *manually* on a ROS graphical interface, Rviz. If the initialization is not good enough then the robot might not reach to the goal location.

The proposed initialization method uses the UWB and LiDAR data. To automate the initialization, the UWB sensors are exploited to estimate the current position of the robot via the source localization algorithm which is explained in Section 2.2.1. UWB data can be only used to find the location information not the heading. However, the location information obtained from them is used as a reference point to match the map data around it and the current LiDAR scans so that the orientation of the robot can be calculated. To achieve this, the LiDAR scan data is rotated ten degrees incrementally and all the points are compared with the map data as shown in Fig. 1 at each iteration. Over the 360 degrees, the distance of each LiDAR end point to the nearest point on the map is calculated repeatedly. The angle value at which the sum of the distances of the points is the lowest gives the orientation of the robot. The pseudo code of the initialization algorithm is given in Table 1.

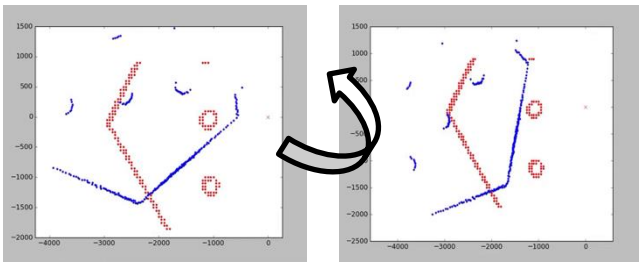


Fig. 1. Ten degrees rotation of LiDAR scan data at each iteration. The blue and red dots correspond to scan and map data respectively.

Table 1. The Pseudo code of the initialization algorithm

Input : M = Map Array , L = Lidar Array

Output : A = angle of robot

1. ***A*** = 0
2. *min_error* = infinity
3. **for** i:=1 **to** 360
4. **if** *i* % 10 = 0
5. ***L[]*** rotate *i* degree
6. *total_distance* = 0
7. **for** *j* = ***L[]*** every element
8. *min_distance* = infinity
9. **for** *k* = ***M[]*** every element
10. **if** *min_distance* > *distance* of *k* to *j*
11. *min_distance* = *distance* of *k* to *j*
12. *total_distance* += *min_distance*
13. **if** *min_error* > *total_distance*
14. ***A*** = *i*
15. *min_error* = *total_distance*

2.3. ROS Ecosystem and System Flow

The location information is the most important information for the navigation systems. The UWB sensors are very good choice for robotic localization applications as they provide range information with low noise components and they are resilient to multipath. Furthermore, the fusion of their information with odometry information leads to have a more robust solution for hard environmental conditions. However, UWB sensors and their fusion with other sensor types and for different indoor applications are not examined well enough in ROS ecosystem. Therefore, in this section the steps of the implementation of ROS modules will be described.

Pozyx UWB sensors (pozyx et. al, 2009) are used through the simulations and the real tests. Therefore, the range information obtained from each UWB anchor in ROS has to be published. However, to determine the robot location, the range information from at least 3 or 4 different sensors for 2D and 3D are needed. Furthermore, sensors may fail randomly, and the erroneous sensor information has to be detected. Thus, a new message type, which contains destination_id array, distance array and stamp array, is created. For each sensor, the message comprises sensor id, range and measurement time and is published as "uwb_data_topic". The codes can be found at the GitHub website (Bostanci et. al, 2009). Additionally, for the simulation purposes, a new UWB simulation node is created and it subscribes to "gazebo/model_states" and gets the real position of the robot on the simulation. After this step, the range information, which is related with the real location, from each sensor is calculated and Gaussian noise is added on it. Then it is shared with "uwb_data_topic" using the same message type. The information from UWB and odometry sensors and *amcl* pose are obtained via "uwb_data_topic" topic, "odom" topic and "amcl_pose topic", respectively. "Odom" topic works with 30 Hz. However, "uwb_data_topic" and "amcl_pose topic" work with 5 Hz and 2 Hz respectively so that every 0.033 s odom information is used to predict the pose and this is verified with UWB data every 0.2 s and *amcl* data every 0.5 s. After these, the pose of the robot is obtained and published with a publisher whose name is "localization_data_topic". The codes can be found in the GitHub website (Bostanci et. al, 2019). Fig. 2 shows the simulation results. The added Gaussian noise is arranged so that the standard deviation is the 1.5% of the true distances (pozxy et. al, 2019). For several trajectory, the Root Mean Square Error (RMSE) of the proposed method is 3cm whereas the RMSE of *amcl* is 6 cm.

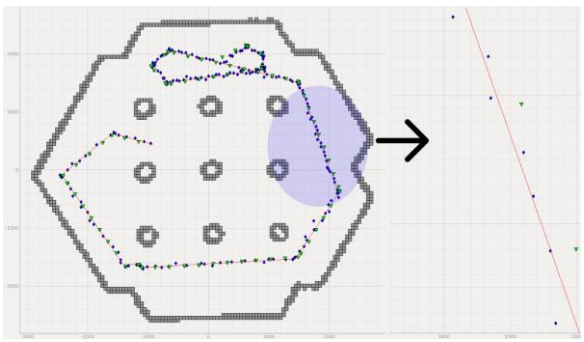
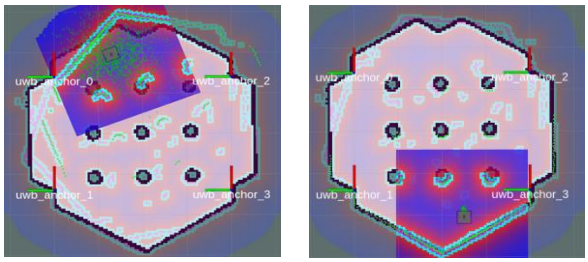


Fig. 2. The blue diamonds, the green triangles and the red line are the estimated trajectory by our method, the estimated trajectory by *amcl* and the true trajectory, respectively.

The source localization algorithm with UWB node provides the location. However, the initial pose also needs an orientation with respect to the map and the system cannot estimate the orientation with only one location. Thus, the map and LiDAR scan matching is applied. After the initial heading estimation, the pose can be published on "initialpose" topic. Fig. 3 shows the pose of the robot before and after the initialization in Rviz. Before the autonomous initialization, the robot position, which is shown with grey rectangle, is wrong and the scan data, which is shown with green dots and lines, is not matched with the real map, which is shown with black dots and lines as demonstrated in Fig. 3a. After the initialization, the scan data and the map are properly matched as shown in Fig. 3b.



a) b)

Fig. 3. a) The robot pose and scans from LiDAR before the automatic initialization. b) The robot pose and LiDAR scans after the automatic initialization.

Lastly, the flow chart of the complete system is shown in Fig. 4. A complete simulation and real test environments are created so that it can be used for different tasks. Robot starts and then the simulation or real-world application is chosen. If the application runs in simulation, the Gazebo simulation and Pozyx simulation are initiated and those two simulators provide the synthetic sensor data and the map data. If the application runs in real world, the real sensor data are obtained. After obtaining either the synthetic data or real data initialization package uses UWB range data and the LiDAR scan data to fullfil the autonomous initialization. Afterwards, *amcl* algorithm takes this information, odometry and scan data to calculate the robot position during the robot trajectory. The Kalman Filter localization node exploits the odometry data to set its motion model and it updates the pose information by using the UWB range measurements and *amcl* pose information. Lastly, the GUI shows the map and the current robot position. If it is needed, the navigation stack can be fed with either with amcl pose information or the pose information obtained from the Kalman Filter which is proven to provide better pose information.

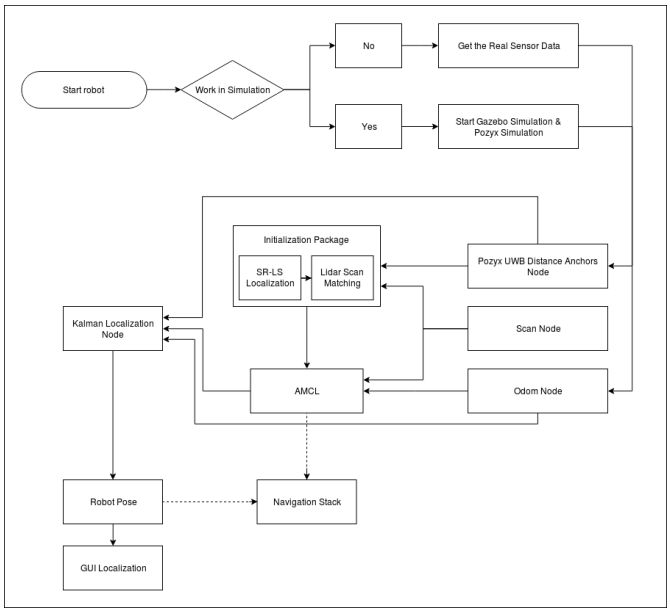
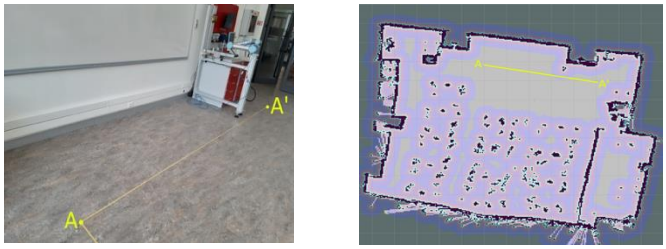


Fig. 4. The flow chart of the complete system

3. Results and Discussion

3.1. Initialization Tests

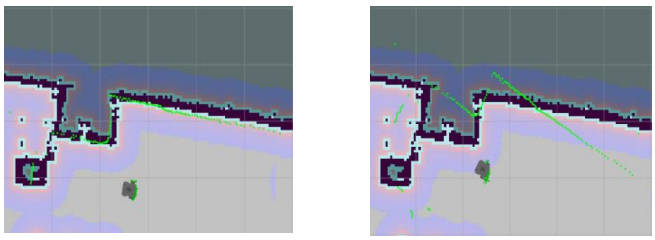
The initialization is essential to have a proper navigation with the navigation stack of ROS. If the robot doesn't know where it is and which direction it is facing properly, the goal it reached might be quite erroneous. The robot is placed on the point A and sent it to the goal point A' for comparison (Fig. 5a). Fig. 5b indicates the map previously obtained from LiDAR data with Gmapping algorithm (Grisetti et. al, 2005).



a) b)

Fig. 5. a) The real test environment. b) The map of the environment obtained with LiDAR.

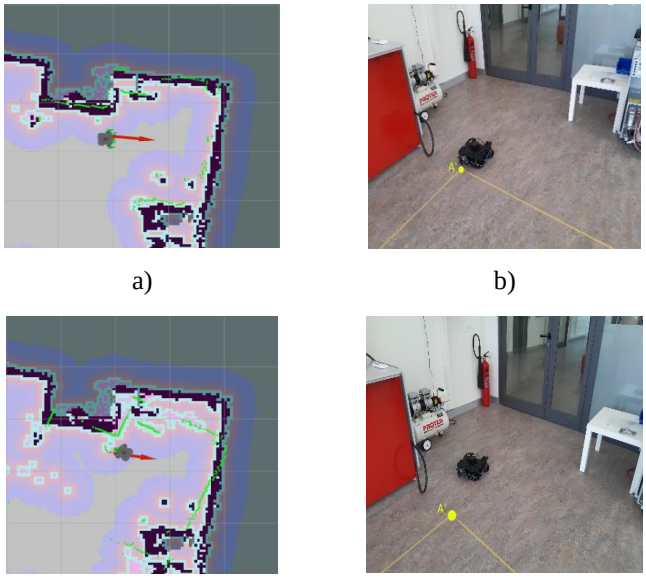
The proposed algorithm uses UWB range measurements to estimate the robot location and LiDAR and map data to detect the direction it's facing. Fig. 6a and 6b show the decently initialized location and the badly initialized location, respectively. The green dotted lines here represent LiDAR scan data on top of the previously obtained map. As can be seen, the Fig. 6a has almost a perfect initializing which means that the position of the robot and its alignment with respect to the real-world match with the map information. On the contrary, the results in Fig. 6b demonstrate the bad matching of the laser data with the map data due to bad location and direction estimation of the robot.



a) b)

Fig. 6. a) The good initialization. b) The bad initialization.

Once the initial pose of the robot is obtained, the robot autonomously goes to A' by exploiting the *move base* algorithm in the navigation stack of ROS. The Figure 7a and 7b shows the final pose of the robot which has started with a good initialization on the map and in the real environment respectively. The robot has perfectly reached the goal as can be validated from the LiDAR scans on the map. The results of bad initialization can be observed in Fig 7c and 7d and the advantageous of having a good estimate on the initial pose is obvious. In other words, the robot is quite far away from the goal and the LiDAR data is not matched with the map at all. Thorough the tests, it is observed that the error above 10 degrees on the initial direction estimate of the robot causes the robot locates in a totally wrong final pose. Therefore, this emphasizes the necessity of the sensitive estimate on the initial pose.



a) b)

c) d)

Fig. 7. The final pose of the robot with a good initialization in a) and b), whereas with a bad initialization in c) and b).

3.2. Localization Tests

For both initialization and localization tests, five UWB anchors are installed in the testing environment which is full of chairs and tables in a 10 m by 10 m area (Fig. 8). The UWB tag on the robot communicates with these anchors to get the range information. Simultaneously, it gets the odometry and *amcl* pose information. Thus, the EKF algorithm fuses these data to get a better pose estimation.

Fig. 8. UWB anchors and the real test environments.

The results are shown in Fig. 9 in which the dimensions are given in mm. The yellow dots represent the UWB anchors while the blue dots and green triangles represent the robot trajectory estimated by the EKF and *amcl*, respectively. The EKF keeps providing the location information even if *amcl* cannot due to the computational complexity. Additionally, the accuracy of the EKF with multiple sensors is better than the *amcl* most of the time. This motivates the usage of sensor fusion that even if one type of sensor measurement is blocked or erroneous, the other type keeps supporting the system.

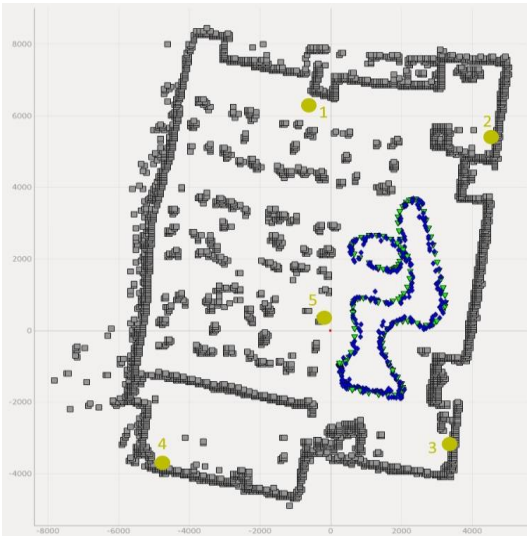


Fig. 9. The estimated trajectory of the robot is shown in blue and green by EKF and *amcl* in real tests respectively.

4. Conclusions and Recommendations

The EKF based localization algorithm, which fuses UWB, *amcl* output and odometry information is proposed for the solution of the localization problem of the robotic applications. The least squares and scan matching based initialization algorithms, which combine the squared range information and scan data, are exploited. The new modules for using UWB sensors for ROS ecosystem have been developed. A complete system flow is created for autonomous robotic applications. The simulation results and the tests in the real environment demonstrate the benefits of UWB sensors both for localization and the initialization of navigation systems in terms of accuracy, the robustness and practicality.

5. Acknowledge

This research is supported by Scientific and Technological Research Council of Turkey (TUBITAK), project number 119E376. I thank Bekir Bostancı, Sercan Tekkök and Emre Söyünmez for their helping me about the real test setup.

References

Açıkel S. and Gökçen A. (2019). Localization and point cloud based 3d mapping with autonomous robots. *European journal of science and technology special issue, pp. 82-92, October 2019.*

Bar-Shalom, Y., Li, X. R., and Kirubarajan, T. (2004). *Estimation with applications to tracking and navigation: theory algorithms and software*. John Wiley & Sons.

Beck, A., Stoica, P., and Li, J. (2008). Exact and approximate solutions of source localization problems. *IEEE Transactions on signal processing*, 56(5), 1770-1778.

Beşkirli, M. and Tefek M. F. (2019). Parçacık sürü optimizasyon algoritması kullanılarak optimum robot yolu planlama. *European journal of science and technology special issue, pp. 201-213, October 2019*.

Bostanci, B., Tekkok, S., Soyunmez, E., and Oguz-Ekim, P. (2019), viewed 17 October 2019, < https://github.com/ieuagv>

Bregar, K., and Mohorčič, M. (2018). Improving indoor localization using convolutional neural networks on computationally restricted devices. *IEEE Access*, *6*, 17429-17441.

Dellaert, F., Fox, D., Burgard, W., and Thrun, S. (1999, May). Monte carlo localization for mobile robots. *In Proceedings of the 1999 IEEE International conference on robotics and automation (ICRA)*, 2, 1322-1328.

Dobrev, Y., Gulden, P., and Vossiek, M. (2018). An indoor positioning system based on wireless range and angle measurements assisted by multi-modal sensor fusion for service robot applications. *IEEE Access*, 6, 69036-69052.

Dudek, G., and Jenkin, M. (2010). *Computational principles of mobile robotics*. Cambridge University Press.

Fox, D., Burgard, W., and Thrun, S. (1998). Active markov localization for mobile robots. *Robotics and Autonomous Systems*, *25*(3-4), 195-207.

Golub, G. and Van Loan, C. (1996). *Matrix Computations*. Johns Hopkins University Press.

González, J., Blanco, J. L., Galindo, C., Ortiz-de-Galisteo, A., Fernández-Madrigal, J. A., Moreno, F. A., and Martinez, J. L. (2009). Mobile robot localization based on ultra-wide-band ranging: A particle filter approach. *Robotics and autonomous systems*, *57*(5), 496-507.

Grisetti, G., Stachniss, C., and Burgard, W. (2005, April). Improving grid-based slam with rao-blackwellized particle filters by adaptive proposals and selective resampling. *In Proceedings of the 2005 IEEE International conference on robotics and automation (ICRA)*, 2432-2437.

Jetto, L., Longhi, S., and Venturini, G. (1999). Development and experimental validation of an adaptive extended Kalman filter for the localization of mobile robots. *IEEE Transactions on Robotics and Automation*, 15(2), 219-229.

Lee, D., Son, S., Yang, K., Park, J., and Lee, H. (2009, August). Sensor fusion localization system for outdoor mobile robot. *In 2009 ICCAS-SICE*, 1384-1387.

Luo F., and Fan Z. (2014). Mobile robot localization based on particle filter. *In Proceeding of the 11th World Congress on Intelligent Control and Automation*, 2014.

Payá, L., Gil, A., and Reinoso, O. (2017). A state-of-the-art review on mapping and localization of mobile robots using omnidirectional vision sensors. *Journal of Sensors*, *2017*.

Pozyx creater kit Lite 2015, viewed 14 October 2019, <https://www.pozyx.io/shop/product/creator-kit-lite-67>.

Robot Operating System (2009), viewed 14 October 2019, < http://www.ros.org>.

Vlassis, N., Terwijn, B., and Krose, B. (2002). Auxiliary particle filter robot localization from high-dimensional sensor observations. In *Proceedings of the 2002 IEEE International conference on robotics and automation (ICRA)*.

Yılmaz Z. and Bayındır L.(2019). Simulation of lidar-based robot detection task using ros and gazebo. *European journal of science and technology special issue, pp. 513-529, October 2019.*

Zhang, L., Zapata, R., and Lépinay, P. (2009, October). Self-adaptive Monte Carlo localization for mobile robots using range sensors. In *2009 IEEE International Conference on Intelligent Robots and Systems (IROS)*, 1541-1546.