2012 Third International Conference on Emerging Security Technologies

EKF based Mobile Robot Localization

Ling Chen, Huosheng Hu, Klaus McDonald-Maier

School of Computer Science and Electronic Engineering
University of Essex, Wivenhoe Park, Colchester CO4 3SQ, United Kingdom
E-mail: {lcheno, hhu, kdm}@essex.ac.uk

***Abstract*—Localization plays a significant role in the autonomous navigation of a mobile robot. This paper investigates mobile robot localization based on Extended Kalman Filter (EKF) algorithm and a feature based map. Corner angles in the environment are detected as the features, and the detailed processes of feature extraction are described. Then the motion model and odometry information are elaborated, and the EKF localization algorithm is presented. Finally, the experimental result is given to verify the feasibility and performance of the proposed localization algorithm.**

***Keywords*-EKF; localization; feature extraction; Odometry; motion model**

I. Introduction

The problem of navigation can be summarized into answering three questions: "where am I?", "where am I going?" and "how should I get there?" [1]. Localization answers the first question "where am I?", finding a reliable solution to this problem paves essential way for solving the remaining two questions.

Since the first mobile robot was invented, the problem of position determination has been investigated and a number of methods have been put forward. These can be classified into two general groups [2]: relative and absolute localization methods. In relative localization, dead reckoning methods such as odometry and inertial navigation are used to calculate the robot position and orientation from a known initial pose. However, their unbounded growth of time integration errors with the distance travelled by the robot is unavoidable [3].

In contrast, in absolute localization, both odometry and external sensor data detecting distinct features of the environment are combined together to estimate the position of the robot. The algorithms used in absolute localization include triangulation and Kalman Filter. One of the disadvantages of triangulation lies in that at least three features or landmarks are needed at the same time to calculate the position, which is impractical in some circumstances, while due to its robustness, Kalman filter is by far the most widely used algorithm for problems in localization, mapping, and navigation [1], [4] . In this paper, an Extended Kalman Filter (EKF) is designed to fuse odometry and laser range information to realize localization.

The rest of the paper is organized as follows. Section II illustrates the principle of extracting the features which are corner angles in this particular project. Then Section III presents an odometry motion model. The design of the EKF localization algorithm is presented in Section IV. Section V provides the experimental result to verify the effectiveness of the EKF algorithm. Finally, a brief conclusion and future work are presented in Section VI.

II. Feature Extraction

In localization, one of the significant steps is to determine the map representation methods. Basically, the methods of map representation can be categorized into three classes: grid map, feature map and topological map. In this paper, the corner angles of the environment are deployed as features and feature map representation is adopted. The process of feature extraction are divided into three steps, which are illustrated in detail as follows.

A. Segmentation

A laser range finder is used to collect the range information of the environment. A full scan of 180°is considered as an ordered sequence of N measurements points(P), where each scanned point can be defined either in Cartesian (x_n, y_n) or in polar coordinates (r_n, α_n), that is:

$$P = \left\{ P_n = \begin{pmatrix} r_n \\ \alpha_n \end{pmatrix} \right\}, n \in [1, N] \quad (1)$$

then one segment S_i can be denoted as

$$S_i = \{(r_j, \alpha_j); (x_j, y_j), j = k : n\}, 1 \leq k < n \leq N \quad (2)$$

Up to now, there are various segmentation methods, some of which are: Point-Distance-based methods (PDBS) [5], [6], Adaptive Breakpoint Detector (ABD) [7], Kalman-Filter-based method [7] and Extended-Kalman-Filter method [8] etc. ADB is chosen as the segmentation method in this paper. By applying each of them to segment the collected laser data sets under the same experimental condition, we can compare the accuracy, robustness and computational efficiency of each method. Based on the comparison, ADB is finally chosen as our segmentation method due to its relatively high accuracy, robustness and less computational expenses.

978-0-7695-4791-6/12 $26.00 © 2012 IEEE
DOI 10.1109/EST.2012.19

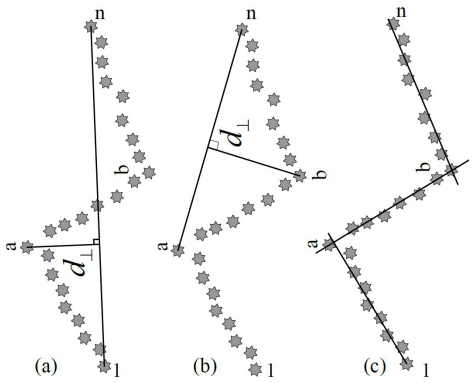

Figure 1. The principle of IEPF

The general principle of ABD is: if $D(r_i, r_{i+1}) > D_{thd}$, then segments are separated, otherwise segments are not segmented, where D_{thd} is the threshold condition and $D(r_i, r_{i+1})$ is the Euclidean distance between two consecutive scanned points

$$D(r_i, r_{i+1}) = \sqrt{r_i^2 + r_{i+1}^2 - 2r_i r_{i+1} cos\Delta\alpha} \tag{3}$$

and the threshold condition is expressed as:

$$D_{thd} = r_i \frac{sin\Delta\alpha}{sin(\lambda - \Delta\alpha)} + 3\delta_r \tag{4}$$

where λ is an auxiliary parameter and δ_r is a residual variance to encompass the stochastic behaviour of the sequence scanned points P_n and the related noise associated to r_n. For this project, we take λ as 10°, δ_r as 10 mm provided by the laser, $\Delta\alpha$ as 1°.

B. Line-Fitting

By line-fitting, we mean estimating the line parameters representing the specific line. Line Tracking (LT) [9] and Iterative End-Point Fit (IEPF) [10] are two classic algorithms for line fitting. Although LT is a very fast and adaptive approach for line extraction, it is very difficult to choose the threshold T_{max}. On the other hand, due to its good performance and being computationally inexpensive, the IEPF and its derivatives are commonly used for line extraction in range images [11]. Therefore, we adopted IEPF as the line fitting algorithm.

The principle of IEPF is to search for a breaking point of a cluster (or segment), which occurs at the maximum perpendicular distance to a line. The process starts by connecting the first and last data points of a cluster via a straight line. The straight line can be formulated as($Ax + By + C = 0$), where $A = y_f - y_s, B = x_s - x_f, C = -Ax_s - By_s$, (x_s, y_s) is the coordinate of the first point, (x_f, y_f) is the coordinate of the last point. Then for all data points between the extreme points, a perpendicular distance $d_\perp$ to the line is calculated as follows.

$$d_{\perp,k} = \frac{Ax_k + By_k + C}{\sqrt{A^2 + B^2}}. \tag{5}$$

If the maximum perpendicular distance $d_\perp$ to the line is greater than a threshold d_{th}, the corresponding point which has the maximum perpendicular distance is determined as the break point. The same process starts by connecting the first data point and the new generated break point, as well as connecting the new generated break point and the last point. This is done recursively until the last point or the maximum perpendicular distance between extreme points is less than the threshold d_{th}. All the points determined by this method are defined as the breaking points of the cluster, including two end points of the cluster. Thus, all the points between two neighboring breaking points are considered as points on the same line. Figure 1 shows the schematic description of IEPF algorithm.

The next step is to determine the line parameters. There are two forms of parameters representing a straight line, the traditional form is:

$$y = mx + q \tag{6}$$

where q and m is the y-intercept and the slope of a line respectively. This form of line representation is called Slope-Intercept form, which is simple and however has one shortcoming, i.e. the vertical lines require infinite m (gradient). When the Slope-Intercept form of parameters is used to calculate the angle of two intersecting lines on a computer, an infinity error will appear.

Fortunately, we have another form of line parameters, i.e. polar form, which is given as follows:

$$\rho = xcos(\theta) + ysin(\theta) \tag{7}$$

where $\rho \geq 0$ is the perpendicular distance of the line to the origin, the angle θ is bounded by $-\pi < \theta \leq \pi$ and is the angle between the x axis and the normal of the line.

This polar form of line representation does not have the shortcoming of Slope-Intercept form. It is more appropriate and accurate to be applied as the form of the line parameters, which will be used as the line parameters here.

We then adopt the line fitting algorithm that was originally proposed by Lu $et\ al.$ [12] to fit the line parameters, which are referred to ρ and θ. Adopting the Least Square regression method, line parameters can be obtained by the following equations:

$$tan(2\theta) = \frac{-2\Sigma(y_m - y_i)(x_m - x_i)}{\Sigma[(y_m - y_i)^2 - (x_m - x_i)^2]} \tag{8}$$

$$\theta = 0.5atan\left(\frac{-2\Sigma(y_m - y_i)(x_m - x_i)}{\Sigma[(y_m - y_i)^2 - (x_m - x_i)^2]}\right)$$

$$\rho = x_m cos(\theta) + y_m sin(\theta) \tag{9}$$

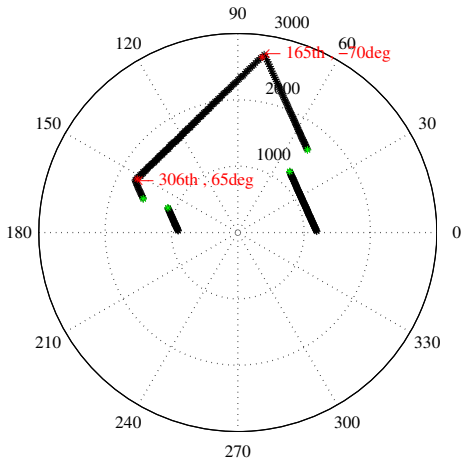

Figure 2. The result of feature extraction

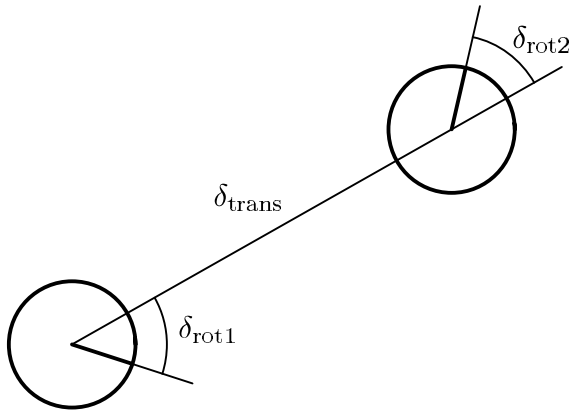

Figure 3. Odometry model: the robot motion in the time interval$(t-1,t]$ is approximated by a rotation δ_{rot1}, followed by a translation δ_{trans} and a second rotation δ_{rot2} [13]

where

$$x_m = \frac{1}{N}\Sigma x_i \ and \ y_m = \frac{1}{N}\Sigma y_i \tag{10}$$

where (x_m, y_m) are the Centroid of the Cartesian coordinates. (x_i, y_i) is the coordinate of each point on the line. N is the number of points in the sector scan we wish to fit line parameter to.

C. Corner Angle Calculation

As long as the line parameters of two intersecting lines are determined, it is easy to calculate the corner angle between these two intersecting lines, i.e. simply by subtracting the parameter θ of one line from that of another line, that is

$$\alpha = \theta_1 - \theta_2 \tag{11}$$

Following all the steps stated in this section, we are able to extract the corner angle feature from the raw laser data. As can be seen in Figure 2, two corner angles are found (the position where two lines intersect) and also calculated.

III. Motion Model

The EKF has two phases, one is prediction phase and the other is update phase. In the prediction phase, the motion model is used to predict the current position of the mobile robot, based on not only the previous estimated position, but also the odometry information such as translational velocity and rotational velocity. According to Thrun $et\ al.$ [13], there are two most widely used forms of motion model for mobile robots, i.e. velocity motion model and odometry motion model.

The velocity motion model has being widely used in the localization and mapping of mobile robots. However, the shortcoming of this model lies in the fact that when the rotational rate ω approaches to 0, the term $\frac{v}{\omega}$ in the model will become infinity, which will result in the calculation error. Therefore, odometry motion model is used in this paper as the motion model in the prediction phase of the EKF algorithm. The detailed illustration is presented as follows.

The motion of a mobile robot during $(t-1,t]$ can be decomposed into 3 steps as shown in Figure 3: the first rotation, then translation and the second rotation.

For most of the commercial mobile robots, it is easy to obtain the odometry information $\bar{x}_{t-1} = (\bar{x}, \bar{y}, \bar{\theta})^T$ and $\bar{x}_t = (\bar{x}', \bar{y}', \bar{\theta}')^T$, which are used to generate the motion model as follows:

$$\delta_{rot1} = atan2(\bar{y}' - \bar{y}, \bar{x}' - \bar{x}) - \theta \tag{12}$$

$$\delta_{trans} = \sqrt{(\bar{x} - \bar{x}')^2 + (\bar{y} - \bar{y}')^2} \tag{13}$$

$$\delta_{rot2} = \bar{\theta}' - \bar{\theta} - \delta_{rot1} \tag{14}$$

$$\begin{pmatrix} x' \\ y' \\ \theta' \end{pmatrix} = \begin{pmatrix} x \\ y \\ \theta \end{pmatrix} + \begin{pmatrix} \delta_{trans} cos(\theta + \delta_{rot1}) \\ \delta_{trans} sin(\theta + \delta_{rot1}) \\ \delta_{rot1} + \delta_{rot2} \end{pmatrix} \tag{15}$$

Equation 15 is the odometry motion model. Practical experience suggests that odometry, although still erroneous, is generally more accurate than velocity motion model.

IV. Extended Kalman Filter Algorithm

Following the successful implementation of feature extraction and motion model, the Extended Kalman filter localization algorithm can be designed and expressed in Table I. The input parameters are μ_{t-1}, Ω_{t-1}, $\bar{x}_{t-1}$, $\bar{x}_t$, z_t, c_t, m, where μ_{t-1} and Ω_{t-1} is respectively the estimated position and the covariance matrix of the position at time $t-1$, $\bar{x}_{t-1}$ and $\bar{x}_t$ are the odometry at time $t-1$ and t respectively. z_t is the observation vector, c_t is the signature of the extracted features, and m is the $a\ priori$ map.

Prediction Step (Lines 1-8): The EKF localization algorithm uses the motion model defined in equation (15). G_t in line 4 is derived by taking the derivative of equation (15)

Table I
THE EKF LOCALIZATION ALGORITHM

Algorithm EKF_localization $(\mu_{t-1}, \Omega_{t-1}, \bar{x}_{t-1}, \bar{x}_t, z_t, c_t, m)$

1: $\delta_{rot1} = atan2(\bar{y}' - \bar{y}, \bar{x}' - \bar{x}) - \theta$
2: $\delta_{trans} = \sqrt{(\bar{x} - \bar{x}')^2 + (\bar{y} - \bar{y}')^2}$
3: $\delta_{rot2} = \bar{\theta}' - \bar{\theta} - \delta_{rot1}$
4: $G_t = \begin{pmatrix} 1 & 0 & -\delta_{trans} sin(\theta + \delta_{rot1}) \\ 0 & 1 & \delta_{trans} cos(\theta + \delta_{rot1}) \\ 0 & 0 & 0 \end{pmatrix}$
5: $V_t = \begin{pmatrix} -\delta_{trans} sin(\theta + \delta_{rot1}) & cos(\theta + \delta_{rot1}) & 0 \\ \delta_{trans} cos(\theta + \delta_{rot1}) & sin(\theta + \delta_{rot1}) & 0 \\ 1 & 0 & 1 \end{pmatrix}$
6: $\bar{\mu}_t = \mu_{t-1} + \begin{pmatrix} \delta_{trans} cos(\theta + \delta_{rot1}) \\ \delta_{trans} sin(\theta + \delta_{rot1}) \\ \delta_{rot1} + \delta_{rot2} \end{pmatrix}$
7: $\bar{\Omega}_t = G_t \Omega_{t-1} G_t^T + V_t M_t V_t^T$
8: $Q_t = \begin{pmatrix} \sigma_r^2 & 0 \\ 0 & \sigma_\phi^2 \end{pmatrix}$
9: **for** all observed features $z_t^i = (r_t^i, \phi_t^i)^T$ **do**
10: $j = c_t^i$
11: $q = (m_{j,x} - \bar{\mu}_{t,x})^2 + (m_{j,y} - \bar{\mu}_{t,y})^2$
12: $\hat{z}_t^i = \begin{pmatrix} \sqrt{q} \\ atan2(m_{j,y} - \bar{\mu}_{t,y}, m_{j,x} - \bar{\mu}_{t,x}) - \bar{\mu}_{t,\theta} \end{pmatrix}$
13: $H_t^i = \begin{pmatrix} -\frac{m_{j,x} - \bar{\mu}_{t,x}}{\sqrt{q}} & -\frac{m_{j,y} - \bar{\mu}_{t,y}}{\sqrt{q}} & 0 \\ \frac{m_{j,y} - \bar{\mu}_{t,y}}{q} & -\frac{m_{j,x} - \bar{\mu}_{t,x}}{q} & -1 \end{pmatrix}$
14: $S_t^i = H_t^i \bar{\Omega}_t [H_t^i]^T + Q_t$
15: $K_t^i = \bar{\Omega}_t [H_t^i]^T [S_t^i]^{-1}$
16: $\bar{\mu}_t = \bar{\mu}_t + K_t^i (z_t^i - \hat{z}_t^i)$
17: $\bar{\Omega}_t = (I - K_t^i H_t^i) \bar{\Omega}_t$
18: **end for**
19: $\mu_t = \bar{\mu}_t$
20: $\Omega_t = \bar{\Omega}_t$
21: **return** μ_t, Ω_t

with respect to x, y, θ respectively. V_t in line 5 is derived by taking the derivative of equation (15) with respect to controls $\delta_{rot1}, \delta_{trans}, \delta_{rot2}$ respectively. M_t in line 6 is the covariance matrix of the noise in *control*, which is referred to $\delta_{rot1}, \delta_{trans}$ and δ_{rot2}. M_t is formulated as

$$M_t = \begin{pmatrix} M_{t(1,1)} & 0 & 0 \\ 0 & M_{t(2,2)} & 0 \\ 0 & 0 & M_{t(3,3)} \end{pmatrix} \quad (16)$$

where $M_{t(1,1)} = \alpha_1 |\delta_{rot1}| + \alpha_2 \delta_{trans}$, $M_{t(2,2)} = \alpha_3 \delta_{trans} + \alpha_4 (|\delta_{rot1}| + |\delta_{rot2}|)$, $M_{t(3,3)} = \alpha_1 |\delta_{rot1}| + \alpha_2 \delta_{trans}$. The parameters α_1 to α_4 are robot-related error parameters, which specify the error accrued with motion [13].

Correction Step (Lines 9-21): Let $j = c_t^i$ be the identity of the landmark or feature that corresponds to the i-th component in the measurement vector. Then we have lines 11 and 12, which is the measurement model. In line 13, H_t^i is the Jacobian of the predicted measurement with respect to the robot location, computed at the predicted mean $\bar{\mu}_t$. Lines 14-18 complete the correction step of the EKF.

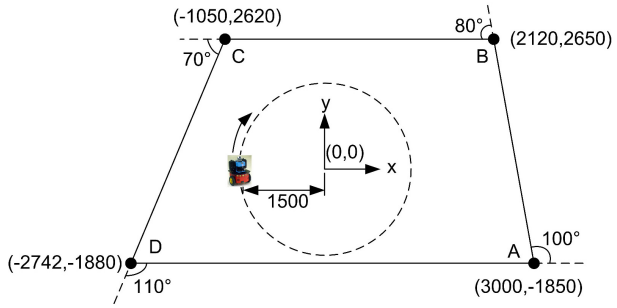

Figure 4. Experimental environment (The unit of coordinates is mm)

V. EXPERIMENTAL RESULT

Theoretically, the application of the EKF localization algorithm will prevent the estimated position from drifting which is the characteristic of using odometry alone. To test the feasibility and effectiveness of the EKF localization algorithm, the mobile robot is programmed to move in a circular trajectory while observing the surrounding environment(shown in Figure 4) with a laser range finder.

As can be seen from Figure 4, the environment is a quadrangle $ABCD$ with four vertexes at which the corner angles can be easily distinguished from each other. If we denote the corner angle by a, the x-coordinate of the vertex point of the corresponding angle by m_x, the y-coordinate of the vertex point of the corresponding angle by m_y, the feature vector F can be given by a collection of triplets:

$$\begin{aligned} F &= \{f_1, f_2, f_3, f_4\} \\ &= \left\{ \begin{pmatrix} a_1 \\ m_{1,x} \\ m_{1,y} \end{pmatrix}, \begin{pmatrix} a_2 \\ m_{2,x} \\ m_{2,y} \end{pmatrix}, \begin{pmatrix} a_3 \\ m_{3,x} \\ m_{3,y} \end{pmatrix}, \begin{pmatrix} a_4 \\ m_{4,x} \\ m_{4,y} \end{pmatrix} \right\} \\ &= \left\{ \begin{pmatrix} 100 \\ 3000 \\ -1850 \end{pmatrix}, \begin{pmatrix} 80 \\ 2120 \\ 2650 \end{pmatrix}, \begin{pmatrix} 70 \\ -1050 \\ 2620 \end{pmatrix}, \begin{pmatrix} 110 \\ -2742 \\ -1880 \end{pmatrix} \right\} \end{aligned} \quad (17)$$

The global coordinates of these four angle points are measured by a VICON tracking system in the robot arena, which can precisely track the objects within the range of the sight of the camera systems.

The mobile robot starts moving from (-1500,0), while the EKF algorithm is executed to obtain the corrected position by fusing both the odometry and the observed feature information. For each time instance (every 100 ms), odometry data can be directly read from the robot. We have the estimated position calculated by EKF as well as the ground-truth position from the VICON system. By comparing these three types of position data, we can verify the effectiveness of the EKF localization algorithm.

Figure 5 shows the experimental data collected from one trial of experiment, providing the odometry, estimated

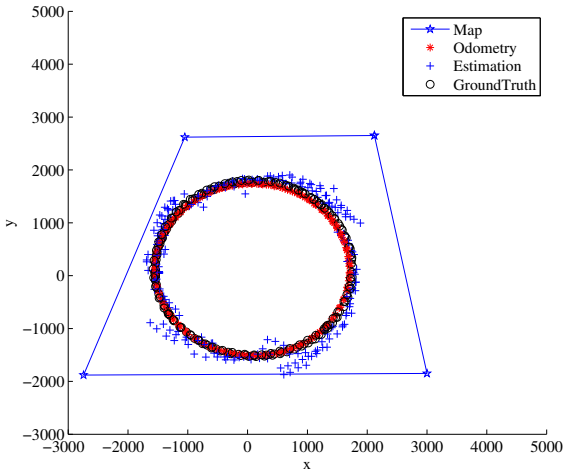

Figure 5. Experimental result: the odometry, estimated position and ground-truth position are shown

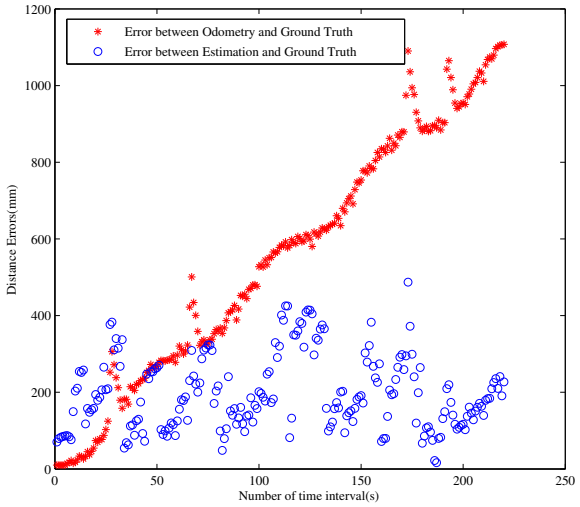

Figure 6. Experimental result: the distance errors

position and the ground-truth position of each instance. Each point in the figure corresponds to an instance in which a measurement is obtained and the filter is updated. However, due to the circular characteristic of the environment, some points may overlap with each other. Since it is difficult to tell apart the three types of position of each time instance, we can not clearly see the effectiveness of the EKF algorithm in this figure.

Therefore, we apply the distance error to better demonstrate the effectiveness of the proposed algorithm. Figure 6 shows the distance error between the odometry and the ground truth respectively. The distance error between the estimated position is calculated from EKF and the ground truth. More specifically, the distance error is calculated as:

$$DEOG_i = \sqrt{(x_{o,i} - x_{g,i})^2 + (y_{o,i} - y_{g,i})^2} \quad (18)$$

$$DEEG_i = \sqrt{(x_{e,i} - x_{g,i})^2 + (y_{e,i} - y_{g,i})^2} \quad (19)$$

where $DEOG_i$ is the ith distance error between odometry and the ground truth. $DEEG_i$ is the distance error between estimated position and the ground truth. $(x_{o,i}, y_{o,i}), (x_{e,i}, y_{e,i}), (x_{g,i}, y_{g,i})$ are the ith coordinates of the odometry, estimated position and the ground truth respectively.

As shown in Figure 6, the distance error between the odometry and the ground truth keep increasing as time goes by, demonstrating that the odometry position is drifting. Nevertheless, the distance error between the estimated position calculated from EKF and the ground truth remains almost the same level, which means that our implementation of EKF algorithm is successful.

VI. Conclusions

This study presents an EKF based mobile robot localization algorithm, in which the detailed process of feature extraction and the odometry motion model are firstly illustrated. The elaborated design and expression of the EKF localization algorithm are then presented. The experiment results are given to verify the feasibility and effectiveness of the proposed EKF localization algorithm.

However, the presented algorithm is implemented in *a priori* map which may not be always available. The future work will focus on the simultaneous localization and mapping (SLAM) using EKF or other more advanced algorithms.

Acknowledgments

The authors gratefully acknowledge the support of the UK EPSRC Global Engagements grant EP/K004638/1 and the EU Interreg IV A 2 Mers Seas Zeen Cross-border Cooperation Programme SYSIASS project: Autonomous and Intelligent Healthcare System (http://www.sysiass.eu/). The 1st author is financially supported by scholarships from China Scholarship Council and Essex University. Our thanks also go to Robin Dowling for his technical support during the research.

References

[1] J. Leonard and H. Durrant-Whyte, "Mobile robot localization by tracking geometric beacons," *IEEE Transactions on Robotics and Automation*, vol. 7, no. 3, pp. 376–382, 1991.

[2] L. F. Borenstein J., H.R. Everett and D. Wehe, "Mobile robot positioning - sensors and techniques," *Jornal of Robotic Systems*, vol. 14, no. 4, pp. 231–249, 1997.

[3] J. Font and J. Batlle, "Mobile robot localization. revisiting the triangulation methods," in *Proceedings of the IFAC Symposium on Robot Control, Bologna*, 2006.

[4] H. Hu and D. Gu, "Landmark-based navigation of industrial mobile robots," *Industrial Robot: An International Journal*, vol. 27, no. 6, pp. 458–467, 2000.

[5] C. Premebida and U. Nunes, "Segmentation and geometric primitives extraction from 2d laser range data for mobile robot applications," *Robótica*, pp. 17–25, 2005.
[6] K. Dietmayer, J. Sparbert, and D. Streller, "Model based object classification and tracking in traffic scenes from range images," in *International Conference on Information Visualisation*, 2001.
[7] G. Borges and M. Aldon, "Line extraction in 2d range images for mobile robotics," *Journal of Intelligent and Robotic Systems*, vol. 40, no. 3, pp. 267–297, 2004.
[8] M. Adams, "On-line gradient based surface discontinuity detection for outdoor scanning range sensors," in *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 3, 2001, pp. 1726–1731.
[9] A. Siadat, A. Kaske, S. Klausmann, M. Dufaut, and R. Husson, "An optimized segmentation method for a 2d laser-scanner applied to mobile robot navigation," in *3rd IFAC Symposium on Intelligent Components and Instruments for Control Applications*. Citeseer, 1997, pp. 153–158.
[10] R. Duda and P. Hart, "Pattern classification and scene analysis," *A Wiley-Interscience Publication, New York: Wiley, 1973*, vol. 1, 1973.
[11] T. Einsele, "Real-time self-localization in unknown indoor environment using a panorama laser range finder," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 2, 1997, pp. 697–702.
[12] F. Lu and E. Milios, "Robot pose estimation in unknown environments by matching 2d range scans," *Journal of Intelligent and Robotic Systems*, vol. 18, no. 3, pp. 249–275, 1997.
[13] W. Burgard, D. Fox, and S. Thrun, *Probabilistic Robotics*. MIT Press, 2005.