Mechatronics 49 (2018) 105–118

Contents lists available at ScienceDirect

Mechatronics

journal homepage: www.elsevier.com/locate/mechatronics

SLAM in dynamic environments via ML-RANSAC

Masoud S. Bahraini[a,b], Mohammad Bozorg[b], Ahmad B. Rad[a,*]

[a] School of Mechatronic Systems Engineering, Simon Fraser University, Surrey, BC V3T 0A3, Canada
[b] Department of Mechanical Engineering, Yazd University, Yazd, 89195-741, Iran

ARTICLE INFO

Keywords:
Autonomous robot
SLAM
DATMO
RANSAC
Multi-target tracking

ABSTRACT

Simultaneous localization and mapping (SLAM) in dynamic environments is an important problem in robotics navigation, yet it is less studied. In this paper, we present a novel approach to segment and track multiple moving objects in real dynamic environments. Detected objects are classified into stationary and moving objects using a state-of-the-art method referred to as multilevel-RANSAC (ML-RANSAC) algorithm. The algorithm is designed to track moving objects in conflict situations while running SLAM. The ML-RANSAC algorithm is developed to robustly estimate velocity and position of the multiple moving objects in an unknown environment whereas the state of the objects (static or dynamic) is not known a priori. The main characteristic of the algorithm is its ability to address both static and dynamic objects in SLAM and to detect and track moving objects (DATMO) without dividing the problem into two separate parts (SLAM and DATMO). We apply the proposed algorithm on two sets of simulated data to validate its performance in situations where the objects are either occluded or placed in dense dynamic scenario. We have compared our method with the true data via simulation studies. Furthermore, we have implemented the algorithm on a Pioneer P3-DX mobile robot navigating a real dynamic environment. Simulation studies as well as real-time experiments suggest that the algorithm is able to track and classify objects accurately while performing SLAM in dynamic environments.

1. Introduction

There have been several successful solutions to SLAM problem in static environments; however, the problem for a dynamic environment is more difficult due to the concurrent requirements of detection, classification, and tracking of moving objects. Therefore, in recent years, SLAM in dynamic environments has attracted the attention of many researchers. Assumption of static objects in a dynamic environment leads to the deterioration of the whole SLAM process due to errors in data association, faults in object detection, failure in loop closing procedure, and erroneous state estimation [1]. These problems can be resolved if the problems of SLAM and DATMO are addressed concurrently. Removing the moving objects from the mapping process leads to improving the quality of the map whereas building a reliable map from static objects results in a successful moving object detection [2]. A survey on SLAM can be found in [3–5]. Also, readers may refer to [6] for a review of DATMO approaches and its applications.

The first remarkable work on SLAM in conjunction with DATMO was presented by Wang et al. [7]. The authors estimated SLAM and DATMO posteriors by decomposing the estimation problem into two separate posteriors for static and dynamic objects. They validated their algorithm using laser range-finder data in an urban environment. Darms et al. [8] investigated a sensor independent algorithm for classification and tracking of dynamic objects using an autonomous vehicle in urban environments. Migliore et al. [9] studied SLAM with moving object tracking in dynamic environments using a monocular camera by applying a MonoSLAM algorithm along with a bearing-only tracker. To reduce the computational complexity, the solution for SLAM was decoupled from the moving object tracking. Other studies [10,11] employed a 2D laser scanners to detect and track moving objects for autonomous vehicles in urban traffic. A bounding box was used to classify the detected objects from the laser data. Lin and Wang [12] studied localization, mapping and moving object tracking in an indoor environment using a stereo camera in dynamic environments. They compared the performance of the stereo SLAMMOT (Simultaneous Localization and Mapping, Moving Object Tracking) and the monocular SLAMMOT and commented that using stereo SLAMMOT improved the performance of SLAM in dynamic environments. The approach, however, suffers from the requirement of a reliable 2D feature tracking over images in order to make the algorithm function properly and the limitation of the single target tracking in an experimental environment. A grid-based localization and local mapping along with DATMO was implemented by Vu et al. [13] in an outdoor environment. A multiple hypothesis tracking (MHT) method coupled with an adaptive

* Corresponding author.
E-mail addresses: ahmad_rad@sfu.ca, arad@sfu.ca (A.B. Rad).

https://doi.org/10.1016/j.mechatronics.2017.12.002
Received 4 May 2017; Received in revised form 18 September 2017; Accepted 11 December 2017
Available online 19 December 2017

interacting multiple model (IMM) filter were used to detect the moving objects. Azim and Aycard [14] proposed a layered approach for classification of moving objects using a 3D range laser in outdoor dynamic environment to reduce the sensor noise and the erroneously detected dynamic objects from a 3D laser scanner data. A transferring occupancy information between two consecutive grids was used by Baig et al. [15] instead of performing a complete SLAM solution in dynamic environment. A 2.5D grid-based DATMO was proposed in [16]. Global nearest neighbor (GNN) data association method and Kalman filtering were used to track moving objects. An EKF based algorithm for a multi-robot simultaneous localization and tracking using MHT was also proposed in [17] by exploiting moving objects in dynamic environments such as the RoboCup and traffic scenarios. The authors mentioned that augmenting moving objects into the localization estimation results in enhancement of localization performance while solving the problem of constructing maps in challenging environments.

Object classification, reliable data association and a consistent tracking algorithm are the main concerns of navigation in a dynamic environment. To achieve a higher degree of accuracy in navigation in such environments, a perfect detection and classification of moving objects are required. A reliable approach for data association prevents updating a track with erroneous measurements, initialization of a false track, or deletion of a real track. Although laser sensors are fast in scanning the surrounding environments and object detection, the classification of detected objects requires considerable time. A robust approach for object detection and classification can help the multi-target tracking (MTT) in a dynamic environment. The segmentation of objects to stationary and dynamic type is the first stage of MTT in a real environment. MTT is still a challenge for researchers when the moving objects are close to each other. A survey of the state-of-the-art results on object detection can be found in [18]. After obtaining a reasonable set of detected features, the next step is the data association, especially in situations where occlusion may happen. The literature [19,20] suggests that at least four types of data association techniques can be applied for SLAM and DATMO: GNN, joint probabilistic data association (JPDA), MHT and RANSAC (RANdom SAmple Consensus). GNN is a simple data association method which is not robust and may fail easily in dynamic environments. JPDA technique works in dynamic environments, but it needs a priori knowledge about the number of targets and sometimes needs hard decisions for data association when there are ambiguities. MHT is the most common method used in MTT algorithms; however, it suffers from computational complexity in the face of exponentially growing number of hypotheses (hypothesis tree) and complexity of implementation [21]. Furthermore, because each step of data association depends on previous steps, unexpected changes in moving objects directions result in losing the object tracking in GNN, JPDA and MHT. RANSAC is one of the most successful approaches to obtain a robust estimation from a dataset which contain both inliers and outliers. It is an iterative method for estimating parameters of a model by constructing model hypotheses from a minimal set of observed data and evaluating the number of measurements that support the hypotheses. The generated hypotheses are compared to obtain a hypothesis with the highest consensus. RANSAC method is robust to a sudden motion change, but faces a difficulty from a randomized selection of all hypotheses in the current frame step. Yang and Wang [22] proposed a specialization of RANSAC algorithm to jointly estimate the position of robot, segment moving objects and detect them. A survey on RANSAC methods can be found in [23,24]. Also, for a survey on recent data association techniques for MTT, see [25].

In spite of the extensive field experiments and research in the area of SLAM, it is still a challenge to navigate an autonomous robot safely in a real dynamic environment. Although there are many algorithms and sensors for detection and tracking of moving objects, choosing the best algorithm and set of sensors is not an easy task. It should be noted that most of the proposed algorithms (cited above) will fail in a pure dynamic environment without any static object and/or they do not support the MTT. The occupancy grid methods for SLAM and DATMO, in general, suffer from high computational cost and are restricted to local mapping. Also, the most of reported methodologies do not support tracking of moving objects with an intermittent observation while running SLAM and are prone to errors in conflict situations. Data association is one of the main problems to track moving objects in a dynamic environment, but a review on the available methods reveals that the traditional data association techniques were used for SLAM and DATMO such as GNN, JPDA and MHT. Applying a 3D laser scanner covers more areas around the robot, but the sensor is more expensive and the classification problem is more complex. Furthermore, supplying a 2D laser scanner on a robot seems more applicable to produce an autonomous robot for indoor applications.

The purpose of this paper is to address some of the problems listed above. We propose the ML-RANSAC algorithm to enhance the speed of the algorithm using the compatibility matrix and updating the state vector in multiple steps. The ML-RANSAC algorithm decreases the number of all generated hypotheses (using a compatibility matrix) in the current frame step in comparison to normal RANSAC algorithm and sequentially improves the performance of data association and estimation. The state of objects is considered to be convertible to static or dynamic throughout the estimation, since we have no information about the type of objects. The main contribution of this paper is on Section 4 where the proposed algorithm tackles SLAM in dynamic environment while doing MTT via the ML-RANSAC algorithm. The proposed algorithm works well in an environment without any static object. The ML-RANSAC algorithm tracks moving objects in conflict situations with an intermittent observation while running SLAM, via robust data association techniques. This algorithm is validated by two simulated and experimental dataset taken from dynamic environments. The framework of the proposed algorithm for SLAM in dynamic environments is shown in Fig. 1 which describes the relation of object detection, ML-RANSAC algorithm, SLAM and moving objects tracking.

The rest of this paper is organized as follows. The theoretical foundation of SLAM and DATMO problems are briefly described in Section 2. A distance-based segmentation technique is described in Section 3 for features detection from raw laser data. The ML-RANSAC algorithm is detailed in Section 4. Section 5 provides the experimental results using the simulation and real data obtained from dynamic environments. Finally, the paper is concluded in Section 6.

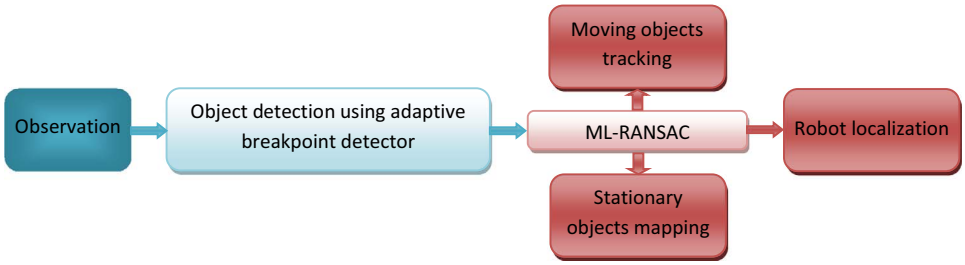

Fig. 1. The proposed framework for SLAM in dynamic environments.

2. Revisiting SLAM and DATMO

SLAM is referred to an ability of an autonomous robot to construct and/or update a map of an unknown environment while simultaneously localizes the robot in the built map. The robot typically starts its navigation from an unknown initial location and moves through the unknown environment in the presence of landmarks. The robot is equipped with one or more sensors to obtain odometry data and observation from the surrounding environment. The details of a SLAM system can be found in [26,27]. As mentioned before, navigation in a real environment in presence of moving objects is still a challenge for autonomous robots. Implementation of a coupled SLAM and DATMO algorithm helps the robot to have a complete knowledge of its surrounding environment. In this section, the kinematic model of a differential mobile robot and the model of stationary and moving objects are described. We assume that detected objects are dimensionless points after object detection process by a laser range finder and collision avoidance is not included in the robot motion controller.

2.1. Nonlinear dynamic model

For a differential drive robot like the Pioneer P3-DX, the kinematic model can be presented as

$$\begin{bmatrix} x_r(k+1) \\ y_r(k+1) \\ \varphi_r(k+1) \end{bmatrix} = \begin{bmatrix} x_r(k) + \Omega_k^r \Delta t \cos(\varphi_r(k) + \Omega_k^d \Delta t) \\ y_r(k) + \Omega_k^r \Delta t \sin(\varphi_r(k) + \Omega_k^d \Delta t) \\ \varphi_r(k) + \Omega_k^d \Delta t \end{bmatrix}, \tag{1}$$

by defining $\Omega_k^r \equiv (\omega_R(k) + \omega_L(k))r/2$ and $\Omega_k^d \equiv (\omega_R(k) - \omega_L(k))r/D$, where r is the active wheels' radius, D is the distance between them, and Δt is the sample time of the discrete fusion process. The state vector of the mobile robot is described by its position (x_r, y_r) and orientation (φ_r). The variables ω_R and ω_L stand for the angular velocity of the right and the left wheels, respectively. The increment of encoder readings can be used to approximate the velocity inputs in this model.

2.2. Feature model

Both dynamic and stationary objects are taken into account in this study. Let us model the state of a stationary object by means of its position and the state of a moving object by means of its position and speed. Therefore, by denoting the location of the ith stationary landmark as X_{si} and representing the jth moving object as X_{dj}, they can be augmented to the robot state vector X_r as

$$X(k) = \begin{bmatrix} X_r(k) \\ X_{s_i}(k) \\ X_{d_j}(k) \end{bmatrix} \in \Re^{3+2n+4m}, \tag{2}$$

where

$$X_r(k) = [x_r(k) \;\; y_r(k) \;\; \varphi_r(k)]^T \in \Re^3, \tag{3}$$

$$X_{s_i}(k) = \begin{bmatrix} \begin{bmatrix} x_{s_1}(k) \\ y_{s_1}(k) \end{bmatrix}^T \begin{bmatrix} x_{s_i}(k) \\ y_{s_i}(k) \end{bmatrix}^T \cdots \begin{bmatrix} x_{s_n}(k) \\ y_{s_n}(k) \end{bmatrix}^T \end{bmatrix}^T \in \Re^{2n},\ i = 1, 2, \ldots, n, \tag{4}$$

$$X_{d_j}(k) = \begin{bmatrix} \begin{bmatrix} x_{d_1}(k) \\ y_{d_1}(k) \\ \dot{x}_{d_1}(k) \\ \dot{y}_{d_1}(k) \end{bmatrix}^T \begin{bmatrix} x_{d_j}(k) \\ y_{d_j}(k) \\ \dot{x}_{d_j}(k) \\ \dot{y}_{d_j}(k) \end{bmatrix}^T \cdots \begin{bmatrix} x_{d_m}(k) \\ y_{d_m}(k) \\ \dot{x}_{d_m}(k) \\ \dot{y}_{d_m}(k) \end{bmatrix}^T \end{bmatrix}^T \in \Re^{4m},\ j = 1, 2, \ldots, m. \tag{5}$$

Note that indices of s_i and d_j stand for the ith static and jth dynamic objects, $x(k)$ and $y(k)$ represent the position of objects in Cartesian coordinates, and $\dot{x}(k)$ and $\dot{y}(k)$ are dynamic object velocities. The movement of the dynamic object jth is defined as a discrete-time dynamic system in Cartesian coordinates (a quite general constant velocity model for motion estimation in horizontal plane) as

$$X_{d_j}(k+1) = T(\Delta t) X_{dj}(k) + w_d(k), j = 1, \cdots m, \tag{6}$$

where X_d is the state of the moving object at time step k described by $X_{d_j} = \begin{bmatrix} x_{d_j} & y_{d_j} & \dot{x}_{d_j} & \dot{y}_{d_j} \end{bmatrix}^T$. In the model given by (6), the acceleration of the track is modeled as noise of process. Matrix $\boldsymbol{T}$ is the transition matrix for the sampling period Δt, and w_d is a zero mean Gaussian process noise with covariance matrix Q_d. The transition matrix and the covariance matrix can be evaluated by

$$T(\Delta t) = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{7}$$

$$Q_d = \sigma_Q^2 \begin{bmatrix} \frac{\Delta t^4}{4} & 0 & \frac{\Delta t^3}{2} & 0 \\ 0 & \frac{\Delta t^4}{4} & 0 & \frac{\Delta t^3}{2} \\ \frac{\Delta t^3}{2} & 0 & \Delta t^2 & 0 \\ 0 & \frac{\Delta t^3}{2} & 0 & \Delta t^2 \end{bmatrix}, \tag{8}$$

where σ_Q is the standard deviation of the process noise.

As mentioned before Δt is the time period between measurements. The final kinematic equation can be augmented as

$$X(k+1) = \begin{bmatrix} F & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & T \end{bmatrix} X(k) + \begin{bmatrix} w_r(k) \\ 0 \\ w_d(k) \end{bmatrix}. \tag{9}$$

Note that the matrix F is the Jacobian matrix which is obtained by linearization of the kinematic model of the robot (Eq. (1)), w_r is the process noise after linearization, and the matrix I is identity matrix.

The Jacobian matrix will be sparse, since the most of the elements are zero. Therefore, using sparse techniques for saving arrays can reduce computational efforts and memory usage.

2.3. Observation model

The observation model can be expressed by a nonlinear function in the form:

$$\begin{aligned} z^{(i)}(k) &= \begin{bmatrix} r^{(i)}(k) \\ \theta^{(i)}(k) \end{bmatrix} \\ &= \begin{bmatrix} \sqrt{(x_m^{(i)} - x_v(k))^2 + (y_m^{(i)} - y_v(k))^2} + v_r(k) \\ \arctan\left(\dfrac{y_m^{(i)} - y_v(k)}{x_m^{(i)} - x_v(k)}\right) - \varphi(k) + v_\theta(k) \end{bmatrix}, \end{aligned} \tag{10}$$

where x_v and y_v are the position of the observation device at time step k. m_i represents the ith feature in the surrounding environment with the pose of $(x_m^{(i)}, y_m^{(i)})$ in the global coordinate X_G–Y_G. The position of ith feature is indicated by $(r^{(i)}, \theta^{(i)})$ with respect to the observation device frame X_R–Y_R. The observation noise ν_r and ν_θ with the standard deviation σ_r and σ_θ are defined for the range and bearing noise, respectively. It should be noted that dynamic objects will be initiated using detected position measured by laser scanner and zero initial velocities. It is assumed that the vehicle is equipped with a device (e.g., laser or radar) to observe the environmental features. In addition, the speed and the relative angle of the robot can be obtained from the encoders of the wheels. Note that there is an offset between the mounting positions of the observation device and the wheelbase of the robot. It can be

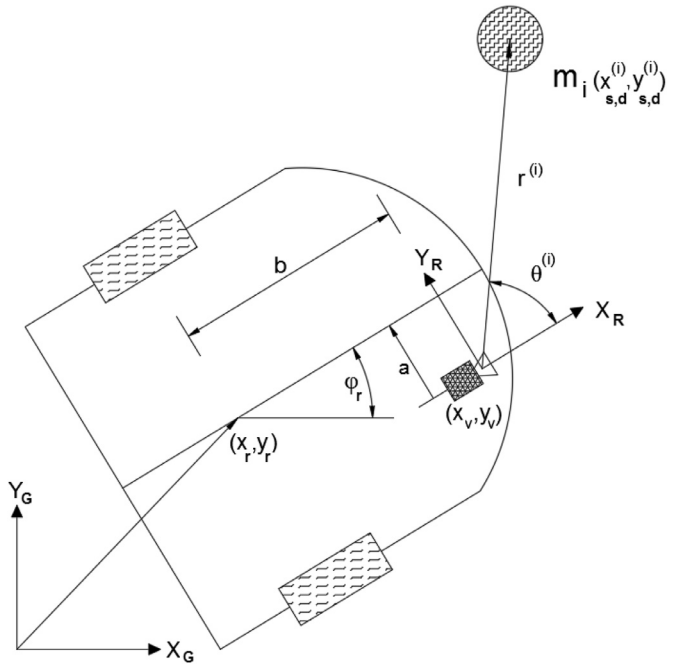

Fig. 2. Vehicle and observation kinematics.

computed by

$$\begin{aligned} x_v(k) &= x(k) + a\cos(\varphi(k)) - b\sin(\varphi(k)), \\ y_v(k) &= y(k) + a\sin(\varphi(k)) + b\cos(\varphi(k)), \end{aligned} \tag{11}$$

where the position of the observation device is defined by parameters a and b with respect to the robot frame. The defined observation in the robot coordinates can be transformed into the global coordinates by a transformation matrix [27]. The basic layout of the observation process and the robot model is presented in Fig. 2.

The measurement model of stationary objects, after linearization of Eq. (10), can be expressed as

$$z(k) = H_s X_s(k) + v(k), \tag{12}$$

where H_s is the observation matrix.

The measurement model of moving objects can be expressed as

$$z(k) = H_d X_d(k) + v(k), \tag{13}$$

where $H_d = [H_s\ 0_{2\times 2}]$, and ν is zero-mean Gaussian noise with covariance R,

$$R = \begin{bmatrix} \sigma_r^2 & 0 \\ 0 & \sigma_\theta^2 \end{bmatrix}. \tag{14}$$

3. Features detection

The available literature reveals that the perception problem and detection of features in dynamic environment are investigated using variety of sensors in different scenarios. We employ a 2D laser scanner to detect features and their position with respect to the robot in a dynamic environment. An overview of moving object detection by laser scanners can be found in [18]. It should be noted that the proposed algorithm requires the range and the bearing of objects from observation device. Therefore, it can be implemented by common observation sensors such as RGBD, monocular camera, stereo camera, laser, and radar. Except the monocular camera, other sensors give us the depth of objects directly. If we want to use a monocular camera, we need to add visual SLAM algorithms to our formulation as well. Also, a distance-based segmentation technique is used as a simple method to find the features in 2D laser data (by searching the differences in range and bearing in polar coordinate, or x and y in the Cartesian case). The proposed ML-RANSAC algorithm does not require a prior knowledge about the type of features to segment them to stationary and moving objects. Therefore, segmentation of features to stationary and dynamical objects is included in the ML-RANSAC algorithm. Laser range-finders with a sufficient resolution and accuracy can be used to navigate a mobile robot in structured indoor environments. In order to use the laser data for navigation, a feature extraction and segmentation method should be applied on raw data. In this paper, a breakpoint detection method based on adaptive threshold is applied to detect features from raw data provided by a conventional 2D laser scanner. Breakpoint detection is an important procedure to cluster features. Breakpoints refer to scan discontinuities in the range image due to the change in range and bearing in the scanning process. The computational effort of main processor and amount of sending data to the main CPU will be reduced by using a decentralized processing method. A micro-controller can be applied to perform a pre-processing stage using segmentation and feature extraction algorithms in real-time application and finally the extracted features will be sent to the main processor. The adaptive breakpoint detector developed in [28] is used to detect and segment features. This algorithm is based on the distance between two consecutive points $[r(i+1), \theta(i+1)]^{\mathrm{T}}$ and $[r(i), \theta(i)]^{\mathrm{T}}$. To separate the laser beam, the segmentation criterion can be stated as

$$\left\| \begin{pmatrix} r(i+1) \\ \theta(i+1) \end{pmatrix} - \begin{pmatrix} r(i) \\ \theta(i) \end{pmatrix} \right\| > r(i).\ \frac{\sin(\Delta\theta)}{\sin(\lambda - \Delta\theta)} + 3\sigma_p, \tag{15}$$

where $\Delta\theta$ is the angular resolution of laser, λ is an auxiliary constant parameter and σ_p is the residual variance to take into account the stochastic behavior of the sequence of the scanned points and the noise associated with the range of laser scanner. In our experimental data, we set the parameters to $\sigma_p = 0.03$ m and $\lambda = 20^\circ$, respectively, which was found to be acceptable. The main interest of using the adaptive breakpoint detector is its simplicity and intuitive appeal along with reasonable performance. The implementation of the adaptive breakpoint detector is described in Algorithm 1. To validate the algorithm, it is applied to one of our experimental dataset. There are some stationary and moving objects in the laser data taken from a real environment. The adaptive breakpoint detector is applied on the laser data. The output results and detected objects are shown in Fig. 3. After obtaining break points, the features will be segmented. The first point and the last point of laser data for a feature can be found from breakpoint detector. Therefore, we can approximate the feature dimension and the angle between them from those points in a set of detected laser points. It should be noted that if we want to detect the breakpoints and separate the laser beams, the distance between two consecutive points must be

Algorithm 1
The adaptive breakpoint detector.

Inputs:
$[r, \theta]^{\mathrm{T}}$ (range and bearing of scanned points), $\Delta\theta$ (the angular resolution), λ (constant parameter) and σ_p (the residual variance)
Outputs:
$[Zr, Z\theta]^{\mathrm{T}}$ (range and bearing of extracted features center)
1. n = number of points
2. **for** $i = 1$ to $n - 1$ **do**
3. $D_{\max} \leftarrow r(i).\ \frac{\sin(\Delta\theta)}{\sin(\lambda - \Delta\theta)} + 3\sigma_p$
4. **if** $\left\| \begin{pmatrix} r(i+1) \\ \theta(i+1) \end{pmatrix} - \begin{pmatrix} r(i) \\ \theta(i) \end{pmatrix} \right\| < D_{\max}$ **then**
   a. feature_points ← $[r(i), \theta(i)]^{\mathrm{T}}$
5. **elseif** feature_points is not empty
   a. feature(k) ← feature_points
   b. k++
6. **end if**
7. **end for**
8. $[Z_r, Z_\theta]^{\mathrm{T}}$ = the middle point of feature points
9. find dimension of features

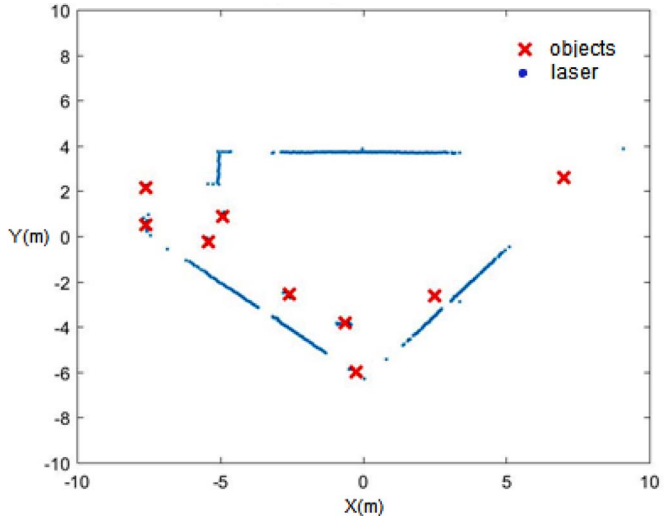

Fig. 3. Real scan example by 2D laser scanner along with detected objects.

greater than $D_{\max}$. The criterion is changed in Algorithm 1, Item 4, because we want to collect the laser beams of a feature. So, we should add the points in a variable (feature_points) until the distance between two consecutive points is less than $D_{\max}$.

4. ML-RANSAC algorithm for SLAMMTT

RANSAC [29] is one of the most successful and widely used iterative algorithms to estimate the parameters of a mathematical model robustly, from a set of (given) observed data in presence of outliers. The main idea of RANSAC is to construct a number of model hypotheses (a number of random samples) from randomly sampled minimal subsets of dataset (observation), and then evaluate the quality of these hypotheses on the entire dataset, where a user specified threshold is required to separate inliers from outliers. The hypothesis with the highest consensus will be selected as the solution. The number of iterations n_{hyp} that is necessary to ensure that a correct solution with probability p is found, can be computed from [30]:

$$n_{hyp} = \frac{\log(1-p)}{\log(1-(1-\varepsilon)^m)}, \tag{16}$$

where m is the minimum number of data points necessary to find estimation successfully and ε is the outliers ratio (percentage of outliers) in the data points. Note that the initial number of iterations is usually chosen high enough (for example 1000), then it will be updated from Eq. (16). The probability p is set to 0.99 in our implementations which means that at least one random sample does not include an outlier. In our framework, the estimated model is the motion of moving objects in a dynamic environment which should be estimated from observation data to search for correspondences or data association. Increasing the number of iterations results in improving the probability quality and growth of the computational effort exponentially.

In summary, the RANSAC algorithm can be stated as two repeated steps: the first step is hypothesis generation which a randomly minimal sample subset is selected from the input dataset to form a set of hypothesis. The parameters of the model are evaluated using only the hypotheses of this smallest sufficient sample subset. The second step is hypothesis validation which the entire dataset is verified to be consistent with the estimated model obtained from the first step. Those hypotheses which lie outside of the estimated model within an error threshold will be considered as outliers.

Algorithm 2
ML-RANSAC.

Inputs:
$\hat{X}^+_{k-1}$, P^+_{k-1} (EKF estimated state and covariance at time step $k-1$), Z_k(measurement at time step k), N (maximum number of iterations allowed in the algorithm), d_1 (first gating area for data association), d_2 (second gating area for data association), d_n (gating area for generating new tracks), Δt (time interval)
Outputs:
$\hat{X}^+_k$, P^+_k (EKF estimated state and covariance at time step k), number of stationary and moving objects
1. **for** each time step k **do**
2. Propagate state estimate and covariance of all states (robot position, static and dynamic features) via EKF.$[\hat{X}^-_k, P^-_k] = prediction(\hat{X}^+_{k-1}, P^+_{k-1})$
3. Search for individual compatibility match using the first gating area d_1
4. Compute the association matrix J, where $J = \begin{cases} 1 & if\ inlier\ (v < d_1) \\ 0 & otherwise \end{cases}$
5. Find the tracks with only one compatibility match in the matrix J
6. First level EKF update for tracks with only one compatibility match, found in the previous step
7. **if** there are remaining observations which need a decision making **then**
   a. find the tracks with the associated feature using RANSAC
   b. $n_{hyp} = N$
   c. **for** $i = 0$ to n_{hyp} **do**
      i. Randomly select estimate and observation matches
      ii. Generate hypotheses
      iii. Only update states using EKF
      iv. Predict all measurements
      v. Compute the hypothesis consensus set
      vi. **if** new hypothesis has larger consensus set than previous hypothesis $(C_n > C_o)$ **then**
         1. Store current hypothesis
         2. $\varepsilon = 1 - \frac{N_{In}}{N_{IC}}$
         3. $n_{hyp} = \frac{\log(1-p)}{\log(1-(1-\varepsilon)^m)}$(updating n_{hyp})
      vii. **end if**
   d. **end for**
   e. second level EKF update with the gating area d_1
8. **end if**
9. (optional) **if** there are remaining observations which need a decision making **then**
   a. find the tracks with the associated observation using the second predefined gating area d_2
   b. third level EKF update (with $d_2 > d_1$)
10. **end if**
11. Prune static objects and moving tracks (static objects are optional)
12. Determine the state of objects (static or dynamic)
13. Initialize new tracks after checking the predefined gating area for initializing new tracks d_n (it can be assumed that all features are moving objects with zero velocity at the beginning)
14. **end for**

4.1. Algorithm description

The proposed ML-RANSAC algorithm is described in detail in this section. The inputs of this algorithm for time step k can be summarized as: EKF estimated state and covariance at step $(k-1)$, set of measurement at time step k, the maximum number of iterations allowed in the algorithm, gating areas for data association, gating area for generating new tracks and the time interval Δt. The kinematic model of the robot, stationary and dynamic objects, and the observation model are given. This algorithm finally returns the outputs: estimated state and covariance (of the vehicle, stationary objects and dynamic objects) at time step k, and the number of stationary and moving objects. The implementation of the ML-RANSAC algorithm can be summarized in Algorithm 2. The algorithm is initialized by setting:

$$\hat{x}^+_0 = E(x_0),\ P^+_0 = E[(x_0 - \hat{x}^+_0)(x_0 - \hat{x}^+_0)^T], \tag{17}$$

at the first time step. Under the assumption of static environment, the state vector does not change during the EKF prediction step, but the state of the moving objects changes in every time step in dynamic

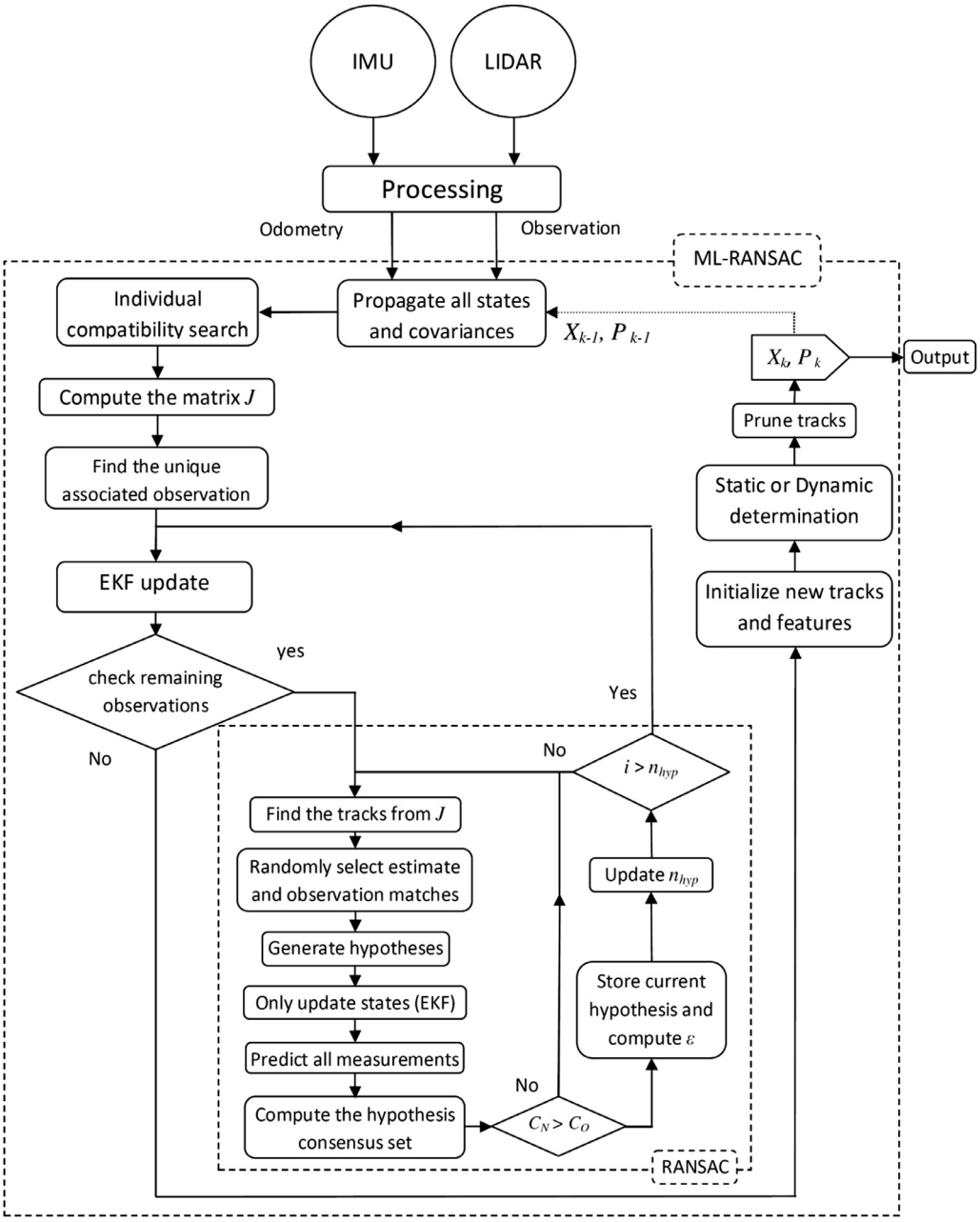

Fig. 4. Overall scheme of the ML-RANSAC algorithm for SLAMMTT.

environments and it has to be augmented to the state of the robot. Taking this into account results in propagation of the state estimates and their covariance from step $(k-1)$ to k, including the robot position and orientation, and the characteristics of static and dynamic objects via EKF (Line 2). Given an initial condition $\hat{x}_{k-1}^{+}$ and initial covariance P_{k-1}^{+}, the propagated states and covariance are given by

$$\hat{x}_k^{-} = f_k(\hat{x}_{k-1}^{+}, u_k), \tag{18}$$

$$P_k^{-} = F_k P_{k-1}^{+} F_k^{T} + G_k Q G_k^{T}, \tag{19}$$

where $x_k \in R^n$ is the state vector at time t_k with an initial condition x_0. u_k is the control inputs to the system at time step k, F_k stands for the Jacobian of f with respect to the state vector $\hat{x}_k^{+}$ at time step k, whereas $f: R^n \rightarrow R^n$ is known as vector function, and Q is the covariance matrix of the process noise. Also G_k stands for the Jacobian of the process noise with respect to the control inputs u_k at step k. Note that $\hat{x}_k^{-}$ is the estimate of x_k before the measurement z_k is taken into account (update step), and $\hat{x}_k^{+}$ is the estimate of x_k, after the measurement z_k is taken into account. The predicted state can be projected into predicted measurements using the known nonlinear measurement equation:

$$\hat{h}_i = h_i(\hat{x}_k^{-}), \tag{20}$$

$$S_i = H_i P_k^{-} H_i^{T} + R_i, \tag{21}$$

where h: $R^n \rightarrow R^m$is known as vector measurement functions, H_i is the Jacobian of the measurement function h_i with respect to the state vector $\hat{x}_k^{-}$, and R_i is the observation noise covariance matrix for the measurement ith assigned to the sensor. Measurements are tested by an active search for individual compatibility. The Mahalanobis distance (ν) is calculated from each observation to track, and then the best observations are selected with smallest distance to each track within a predefined validation gate (d_1) (Line 3). The association matrix J is established to ensure the geometric compatibility of features. It states the binary relation of the measurements to the existing tracks without disregarding the ambiguous associations. The value of one and zero in the association matrix J stand for the measurement either is an inlier or

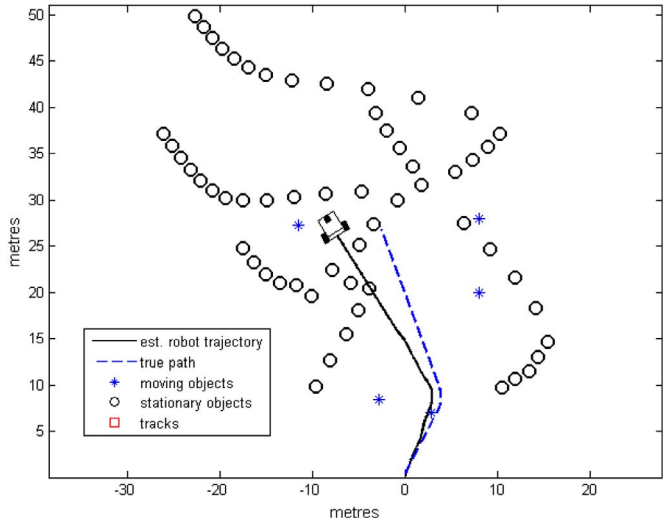

Fig. 5. Applying SLAM algorithm in dynamic environment. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

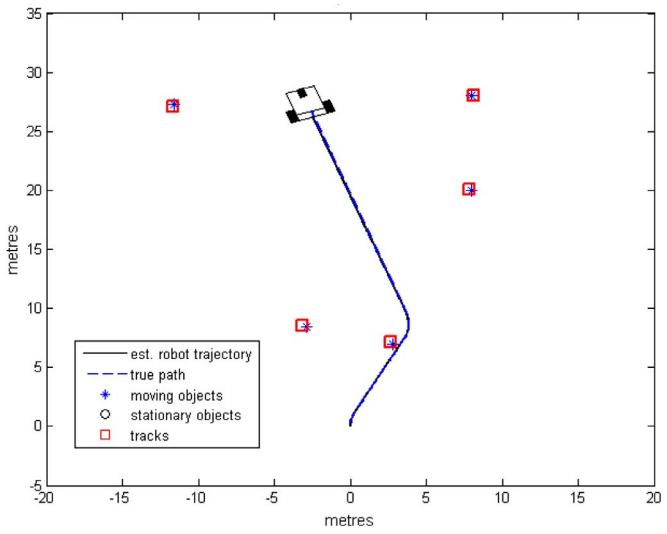

Fig. 6. Applying proposed SLAM and DATMO algorithm in dynamic environment. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

is not, respectively (Line 4). Using the association Matrix J, we can generate reasonable hypotheses which can be happened in reality, instead of generating all possible hypotheses. Applying this technique results in reducing computational efforts, due to determination of the appropriate matching between the estimated tracks and measurement features in the RANSAC part. If the measurement z is an inlier to only one track, then it is used to update the associated track according to the association matrix J in the first level (Lines 5 and 6). The measurement update of the state estimate can be performed using the normal Kalman filter equations:

$$\begin{aligned} K_k &= P_k^- H_k^T S_k^{-1}, \\ \hat{x}_k^+ &= \hat{x}_k^- + K_k(z_k - h_k(\hat{x}_k^-)), \\ P_k^+ &= (I - K_k H_k) P_k^-, \end{aligned} \tag{22}$$

where H_k stands for the augmented Jacobian of all measurements and $h_k(\hat{x}_k^-)$ transforms the features positions into the sensor coordinate. So until here we have not used the RANSAC iteration which can take more computational time. After updating those tracks which are an inlier to only one measurement, some observations might remain without association with any track. When multiple observations are received around a track, linking this track to a real observation requires a hard decision making. So, if there are some other observations which can be associated with more than one track, the algorithm finds the best solution to make a decision for those observations using the RANSAC algorithm (Line 7). Otherwise, we skip the RANSAC algorithm to save computational efforts. Then, n hypotheses are randomly generated from observation data and the estimated tracks. Using these hypotheses, the states are updated using EKF formula. In the next step all measurements are predicted to compute the hypothesis consensus by counting measurements inside a threshold. At the end of this part, the hypothesis will be compared with the previous ones. If the new hypothesis has larger consensus than the maximum consensus of the previous hypotheses, it will be stored as the best hypothesis. It should be noted that the random hypotheses will be created based on individual compatibility data as well as the predicted state in the current algorithm. After receiving the association information, it is going to associate the previously gathered data to implement the measurement update (Line 7.e). If there is still another observation which needs a decision making, it could be found by the second predefined gating area (d_2) which is an optional step (Line 9). In an environment with high density of moving objects, this step can help the algorithm to make hard decisions, smoothly. If a track does not have any detection for a continuous number of scans, so it should be deleted (Line 11). Some of stationary objects can be converted to moving object and vice versa in real world, so the algorithm should be able to handle this transformation as well. The status of objects can be changed by checking the position of them while performing SLAM and DATMO (Line 12). If a measurement in the association matrix J is an outlier to all existing tracks, then it is used to create a new track after checking the predefined gating area for initializing new tracks d_n (Line 13). This implies that if there is a column with all zero elements in the association matrix J, this measurement is an outlier to all existing tracks. A gating area is set for initializing new tracks ($d_n > d_2$) to ensure that the measurement cannot be associated with other tracks. An overall scheme of the ML-RANSAC algorithm for SLAMMTT is illustrated in Fig. 4. A LIDAR sensor is employed to obtain the observation of features in the surrounding environment. The odometry data can be obtained from IMU and encoders of the robot. As it is shown in Fig. 4, propagating the states and covariance, constructing the association matrix J and the measurement updating of the estimates at the first level can be followed step by step in Algorithm 2. Some observations might be associated with the existing tracks, so we need to check the remaining observations in this step. If there is not any other observation, we bypass the RANSAC iterations to save computational efforts. Else if there are still some observations which should be associated with tracks, the RANSAC algorithm gives us the appropriate matching between the estimated tracks and observations. If a measurement is an outlier to all existing tracks, it can be defined as a new track by converting the relative measurement to the global coordinates in the built map. Tracks and stationary objects should be managed in the map by initialization of new tracks or new stationary objects, converting static objects to moving objects or vice versa and deleting old tracks. The status of objects can be detected by checking any change in their positions.

5. Results and discussion

To evaluate the performance of the proposed ML-RANSAC SLAMMTT approach, a series of simulation studies were conducted. Additionally, extensive experiments were performed by the mobile robot Pioneer P3-DX in an indoor dynamic environment. All experiments demonstrate that our approach can accurately and reliably estimate the robot pose, construct the map, and keep tracking of moving objects even in situations of occlusions in which the robot is moving at speed of up to 2 m/s.

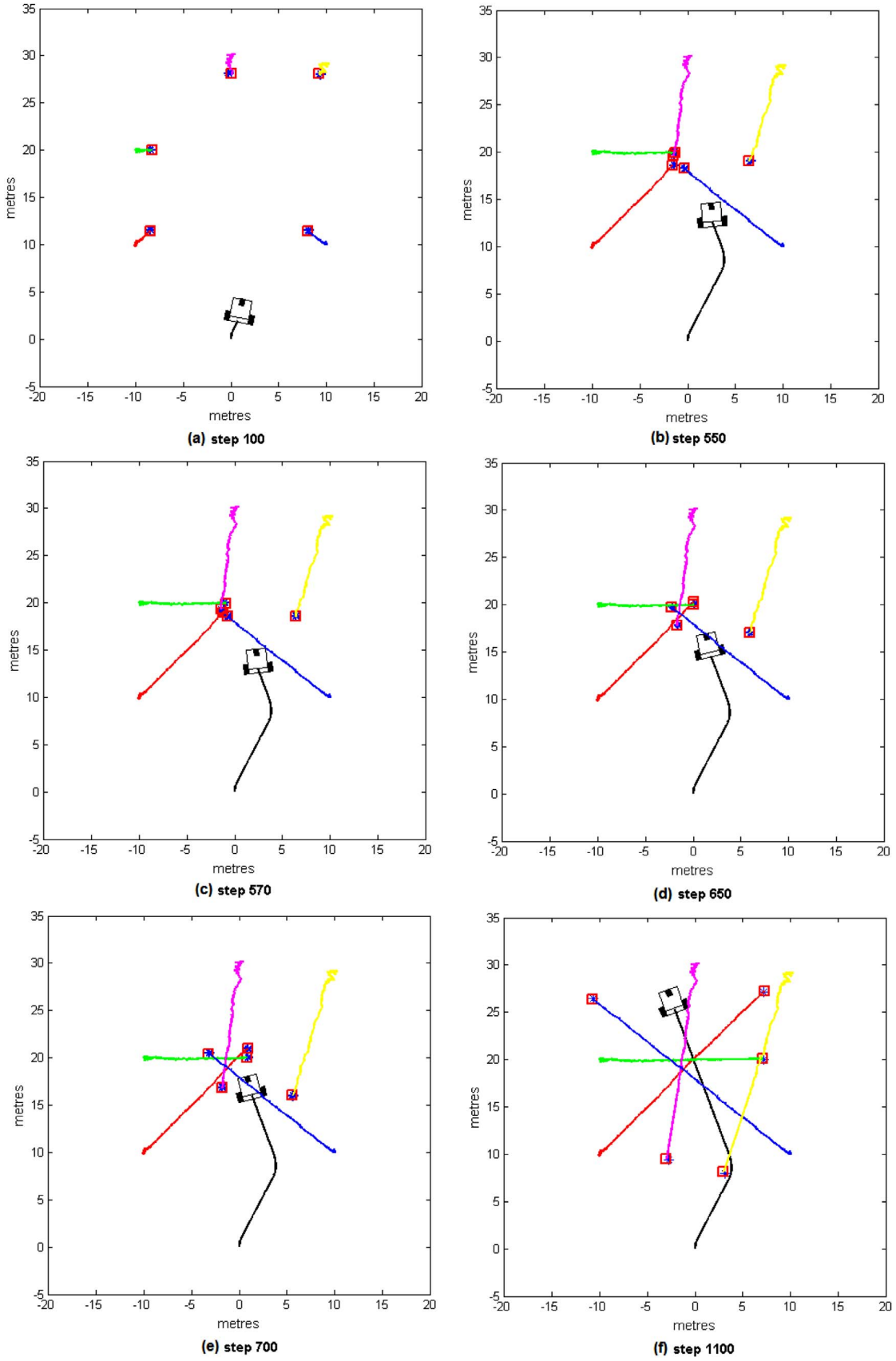

Fig. 7. The estimated trajectories of the robot and moving objects in the selected steps. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

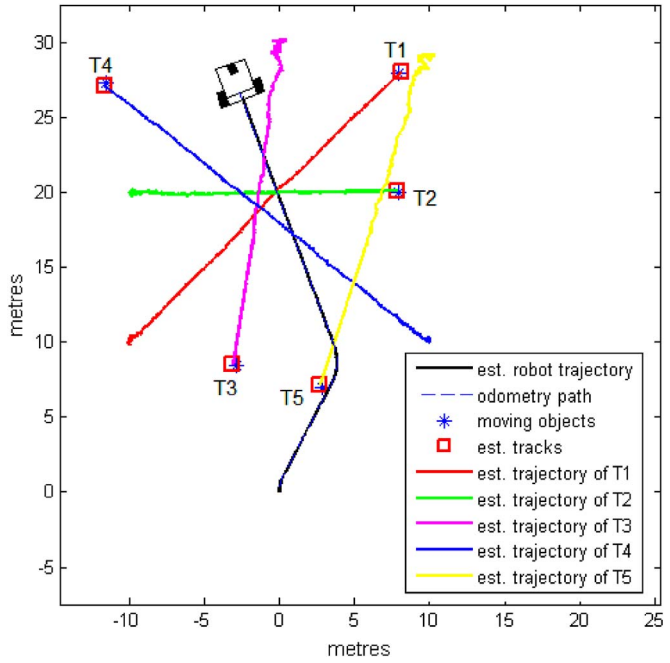

Fig. 8. The estimated trajectories of the robot and moving objects (color continuous lines). (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

5.1. Simulation results

In order to show the efficiency and robustness of the proposed algorithm, the simulation results were carried out in two different scenarios. The first scenario was performed in proximity situations for 5 moving objects in an indoor environment, whilst another scenario was carried out in a complex indoor environment including both stationary and moving objects.

In the first experiment, the simulated robot was moving with speed of up to 2 m/s in a dynamic environment. Simultaneously, five moving objects were moving in this environment (shown as blue asterisk in Figs. 5 and 6). It was assumed that the positions of the objects were captured with respect to the robot by a laser range finder with maximum distance of 30 m and field of view 180°. During all of the simulation experiments, the observation noises were set to 0.2 m and 2° for range and bearing, respectively. The update frequency was eight scans per second for the laser range finder. In this simulation experiment, the tracks are moving with constant velocities while the estimator does not have information about them. If the estimator does not have any observation from features for a certain period of time (which is 3.5 s in our current system), the track will be removed from the track list.

Fig. 5 shows a typical example of SLAM in dynamic environment which the restriction of using only SLAM leads to a wrong localization and mapping. Herein, the robot moves through an unknown environment including several moving objects (shown by asterisk) which move with unknown velocity towards each other in the vicinity of the mobile robot. The circles indicate the estimated position of the moving objects, which are obtained by applying EKF SLAM on the dataset. It was observed that we could not construct a true map from environment and the algorithm diverged. The algorithm mapped the moving objects as stationary objects, wrongly. Also, the estimated path computed by the EKF along with the true path is shown in Fig. 5. Herein, the dashed line indicates the true path, whereas the estimated trajectory of the robot is indicated by a black continuous line. This scenario demonstrates the fact that we need a different algorithm to track moving objects. The ML-RANSAC correctly localizes the robot and concurrently tracks the moving objects which pass through the environment. This is indicated by a comparison of the two approaches in Figs. 5 and 6. Thus the ML-RANSAC approach generates more accurate localization than methods using only SLAM algorithms. The estimated path of the robot and the map of the environment are illustrated using the proposed ML-RANSAC algorithm via EKF filter. To analyze the advantage of the explicit proximity situations handling in the proposed algorithm, we constructed a dataset with situations of proximity. The estimated trajectory of the robot along with the estimated trajectories of moving objects (colored continuous lines) are plotted in Fig. 7. This figure shows a particularly challenging situation in which the ML-RANSAC algorithm is able to successfully track several people (moving objects) walking with velocities of ($V_x(T_1) = 0.625$, $V_y(T_1) = 0.625$, $V_x(T_2) = 0.625$, $V_y(T_2) = 0$, $V_x(T_3) = -0.1$, $V_y(T_3) = -0.75$, $V_x(T_4) = -0.25$, $V_y(T_4) = -0.8$, $V_x(T_5) = -0.75$, $V_y(T_5) = 0.6$) along x and y directions whereas temporarily occluding each other. The estimated position of the robot and tracks are plotted in the selected steps in Fig. 7. It can be seen that the moving objects are tracked correctly through the motion. The proximity situation can be observed in Step 550–650, clearly. Although the dynamic objects are moving toward each other, the estimator keeps tracking of objects continuously while the mobile robot is moving and localizing itself. Additionally, the final estimated trajectories of moving objects and the robot are plotted in Fig. 8. The trajectory error of the tracks are plotted in Fig. 9. It can be seen that losing the observation of features (for example from step 700 for track T3 (violet)) leads to increasing the error of estimation. Although losing observation leads to increasing error in tracking process, the error of

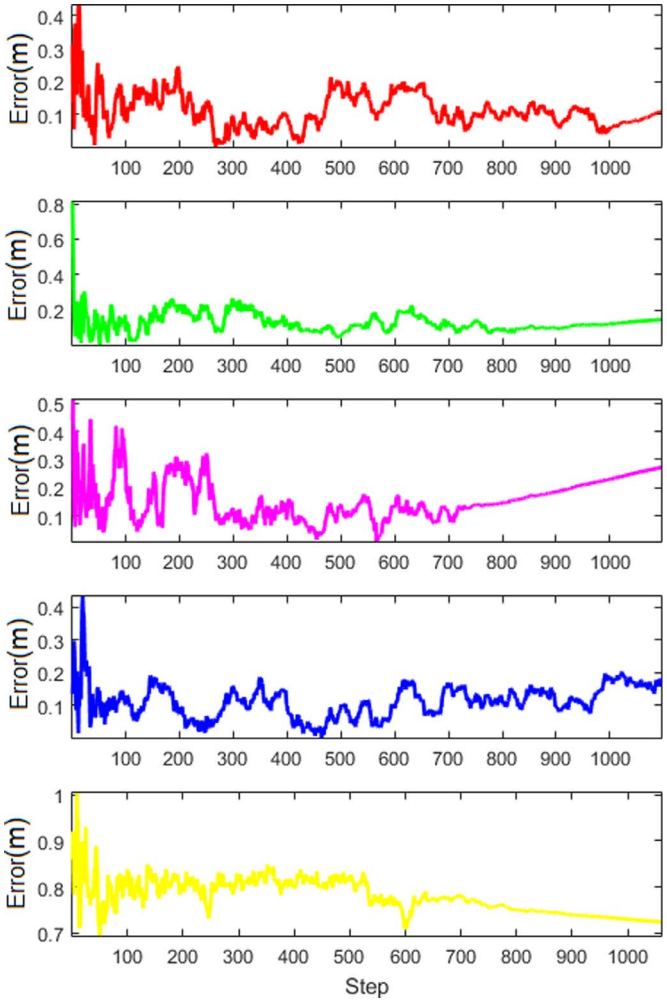

Fig. 9. The mean square error of estimated trajectories of the moving objects (T1–T5 from up to down). (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

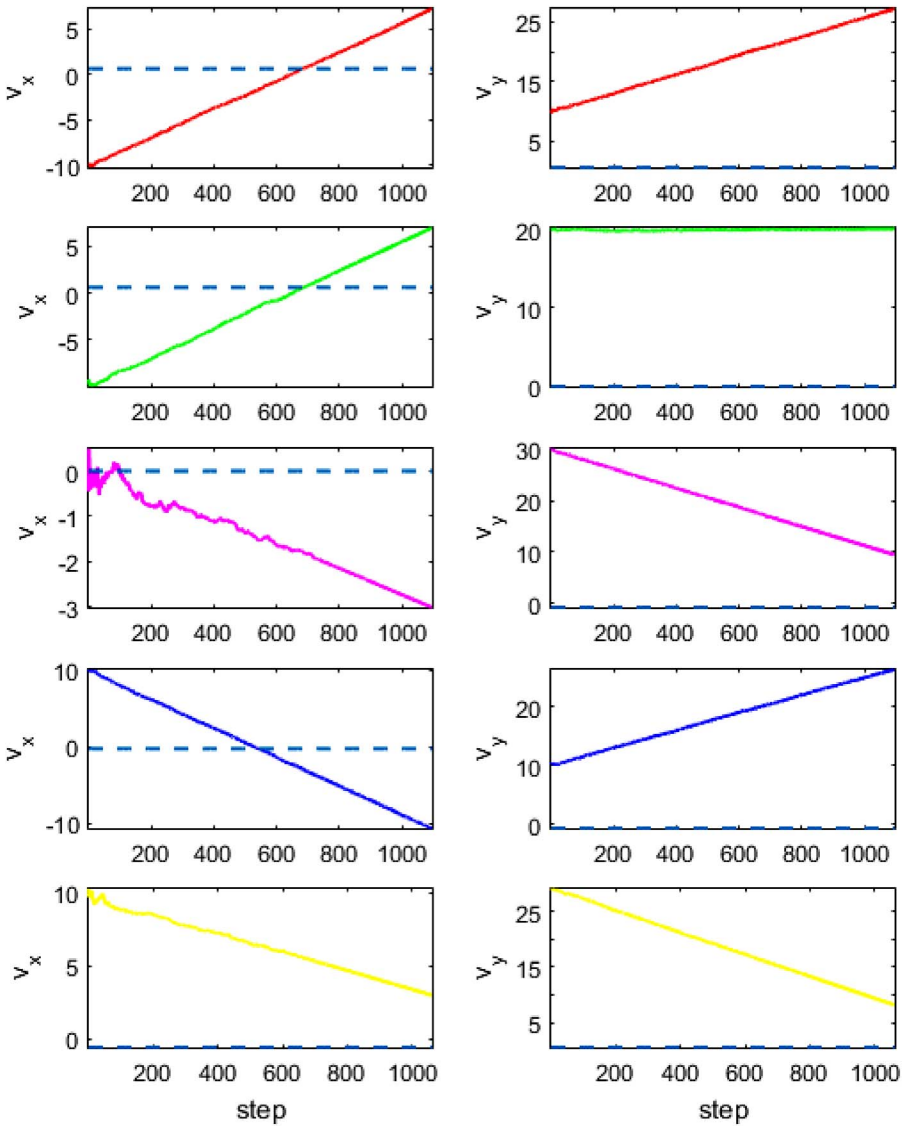

Fig. 10. Velocities of tracks in each step (T1–T5 from up to down). (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

the estimated trajectory is still acceptable using the proposed algorithm. Fig. 10 shows the estimated velocities of tracks in each step through the motion. The colored lines in this figure correspond to each track in Fig. 8 from T1 to T5. The velocity error was going to zero for some tracks and then it was increased. The reason is due to losing observation when those tracks were deemed to be out of the observation area since the assumed laser scanner in the simulation studies have had 180° field of view. To improve the target tracking in a real environment, we have used a laser scanner with 360° field of view in our experiment in real dynamic environment.

In the second experiment, the simulated robot was moving with speed of up to 2 m/s in a dynamic environment like the lobby of a building, where some stationary objects existed. Simultaneously, several moving persons were walking in this area, with frequently changing their orientation of motion. Continuously, the types of objects were classified and tracks were detected as moving objects. The robot was planned to move in the illustrated path while the moving objects walked around the robot and entered to the environment or exited from it. Fig. 11 shows the robot and the dynamic environment. It can be seen that using the proposed algorithm in the proximity situations, the system constructed the map of the stationary objects and tracked the moving objects in a reliable manner. Thus, the ML-RANSAC algorithm improves the performance of the system. It should be noted that most of tracking algorithms will fail, if one of the two sample sets is removed or if one sample set tracks the wrong moving object after the proximity took place. The performance of our tracking algorithm is evaluated with and without proximity situation through the motion of the robot in the described environment.

The estimated position of the robot and tracks are plotted through the motion by the selected steps in Fig. 11. Here, the robot navigates an unknown environment with several stationary and moving objects which are indicated by asterisk and plus signs, respectively. The black circles red squares indicate the estimated position of the stationary and moving objects, respectively. Fig 11(a) and (b) shows that the tracks are kept continuously, even by 180° changing the orientation of moving objects. It should be noted that in the situations like this, algorithm generates new tracks from moving objects and the old tracks will be deleted after several steps because we do not receive observation from them anymore. Fig 11(c)–(f) shows some particularly challenging situations in which moving objects temporarily occluding each other or even stationary objects to receive observation from features. It can be seen that the proposed algorithm is able to successfully track moving objects and to estimate position of stationary objects. Also, the estimated trajectory of the robot in the indoor dynamic environment is given in Fig. 12 whereas it is compared with the true path of robot and the odometry data.

The described experiments demonstrated that the proposed algorithm is able to reliably provide accurate estimate of the robot position,

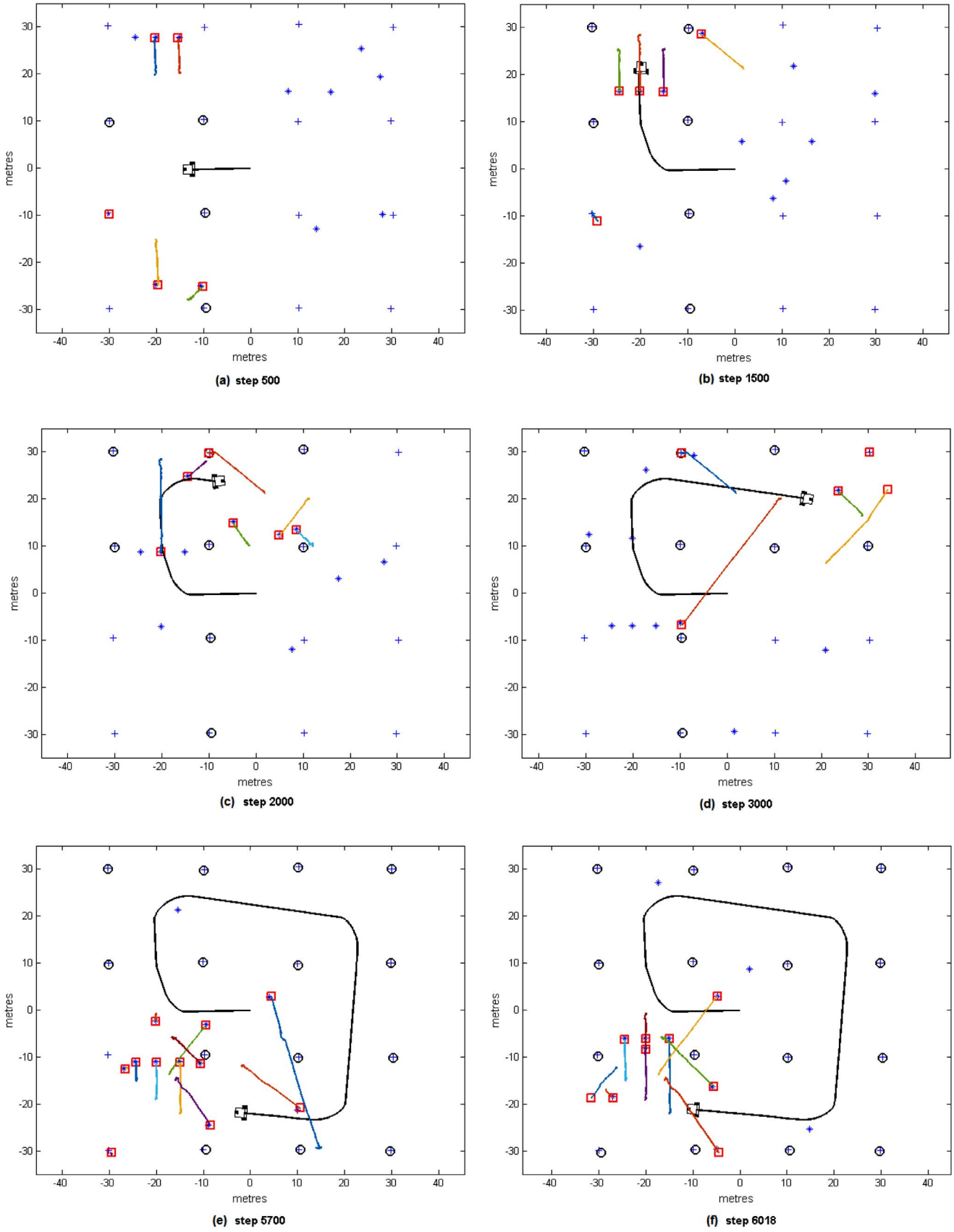

Fig. 11. The simulation results of SLAMMTT procedure in selected time steps for a complex indoor scene. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

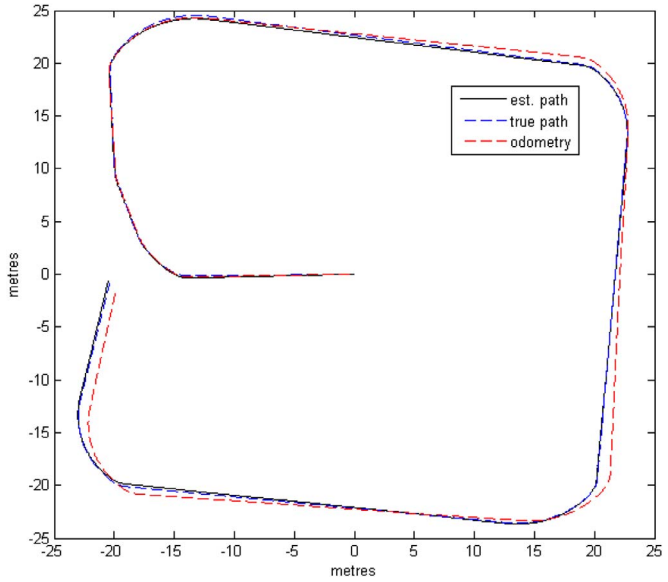

Fig. 12. The estimated and true trajectory of robot in the indoor dynamic environment.

to construct the map of environment, and to track several moving objects even in presence of proximity situations. Note that during all of our experiments the type of objects (static or dynamic) is unknown for the system and estimator. Additionally, the proposed method is able to reliably keep tracking of both long and short observed moving objects.

5.2. Experimental results (indoor scene)

The test bed for experimental studies was a Pioneer P3-DX mobile robot augmented with sensors as shown in Fig. 13. Although different types of range finder sensors were installed on the robot, such as sonar, stereo camera and RGBD, LIDAR (RPLIDAR), we only obtained dataset and validate the proposed algorithm using RPLIDAR.

A 360° RPLIDAR laser scanner was mounted on the top of the mobile robot. The RPLIDAR (a2) is a low cost 2D laser scanner with range of 8 to 16 m, precision of 2 mm, scan frequency of 10 Hz and resolution of 0.9°. The 360°' laser scanner covers the significant area surrounding the mobile robot which is appropriate for indoor environment. Applying LIDAR for observation, leads to decreasing the computational effort and sequentially, expedite the detection of features in occluded situation, classification of objects and accuracy in position estimation.

The results of detection and tracking of moving objects task for indoor experiment are presented in Fig. 14. In this experiment, three walking persons were correctly tracked in the occlusion situation. The speed of the robot is 25 cm/s. Note that the current speed of the robot was chosen according to the frequency of the observation device. If we increase the speed of the robot, we need an observation device with higher frequency. Increasing the observation frequency results in increasing computational efforts. We can easily increase the speed of the robot in simulation studies to the same speed of moving objects or even more, since there is possibility of increasing the frequency of observation. In simulation studies, the robot was moving at speeds of up to 2 m/s, but there is a limitation for hardware and practical applications. The moving objects (pedestrians) performed a normal walking with a constant velocity around 150 cm/s. Although there is no observation from velocities of moving objects (there is a restriction on gathering data from velocity) and the measured information from sensor was included uncertainty about the position of objects, the proposed algorithm shows a considerable robustness to keep tracking of moving objects and estimate the trajectory of the robot. Although, we have had an intermittent observation from one of the moving objects between two Fig. 14(a) and (b), the proposed algorithm kept tracking of moving object. From Fig. 14(c), it can be seen that the tracking of moving objects are kept even behind of the robot by exploiting a 360° laser scanner.

Fig. 13. Pioneer P3-DX mobile robot with mounted sensors.

6. Conclusion

In this paper, an algorithm is proposed to include moving objects in an SLAM formulation. The rationale for this study is noting that many SLAM algorithms fail when the observation is not continuous during the tracking process [31]. Also, most SLAM algorithms for dynamic environments require the static objects as essential components to stay localized and to construct a map from the surrounding environment while tracking the moving objects. Detection and classification of objects are important problems when a robot is moving through a dynamic environment. Data association is another important problem for SLAM in dynamic scenes. The contribution of the proposed ML-RANSAC algorithm is to alleviate the main drawbacks of SLAM in presence of moving objects. To validate the proposed ML-RANSAC SLAMMTT approach in presence of moving objects, a series of simulation studies were conducted in challenging dynamic environments such as an environment without static objects and dense dynamic environments in presence of proximity situations, occlusion situations and

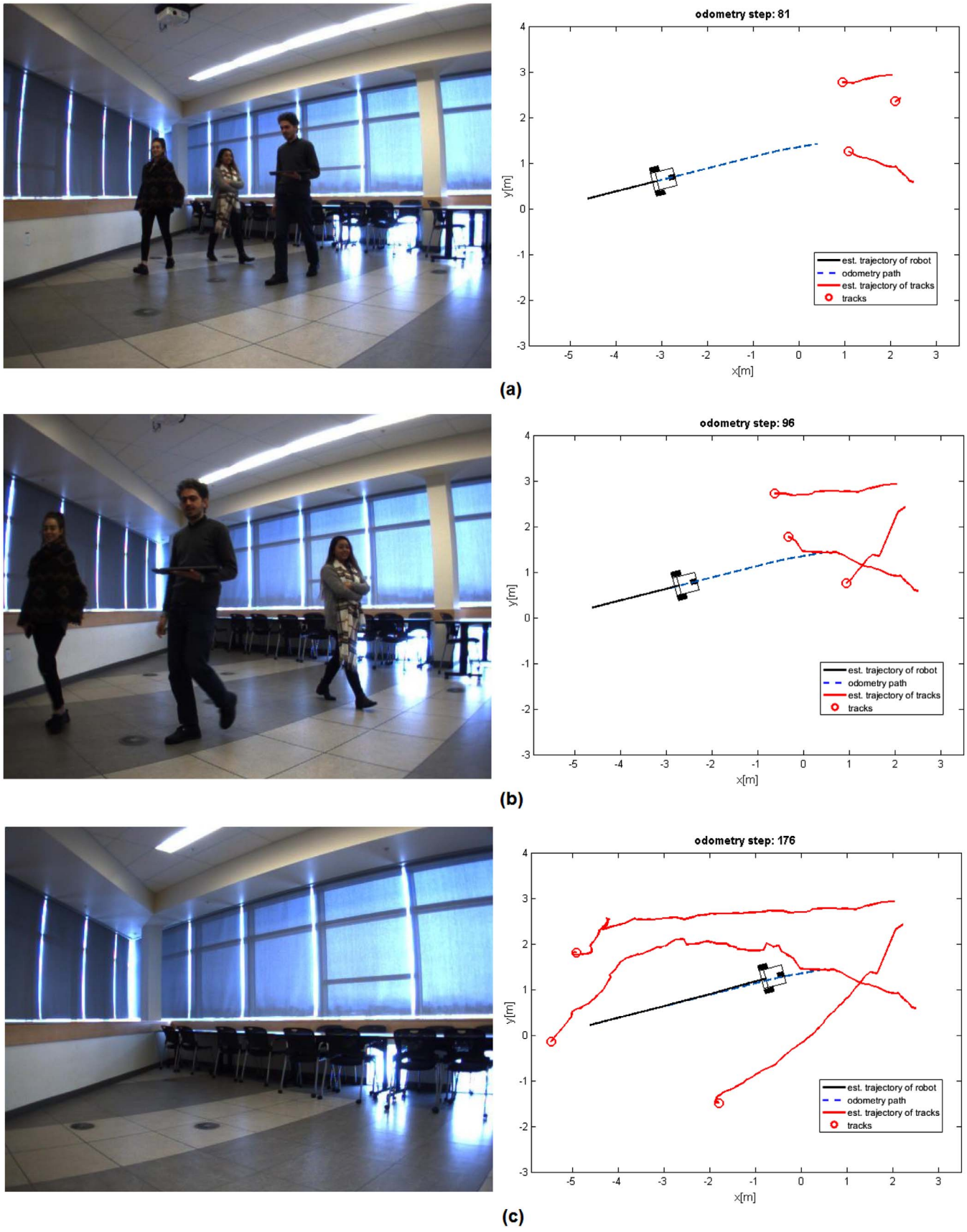

Fig. 14. The estimated trajectories of the robot and moving objects.

intermittent observations. The proposed ML-RANSAC algorithm works well and maintains tracking of moving objects even by receiving intermittent observations or when there is not any static object in the environment. Simulation studies as well as experimental results confirm the effectiveness and feasibility of the algorithm in challenging environments. Simulation and experimental studies have verified the performance of the proposed algorithm even for the cases where an unknown number of randomly placed moving objects exist in a dynamic scene.

References

[1] Zamora E, Yu W. Recent advances on simultaneous localization and mapping for mobile robots. IETE Techn Rev 2013;30:490–6.
[2] Wang DZ, Posner I, Newman P. A new approach to model-free tracking with 2D LIDAR, robotics research. Springer; 2016. p. 557–73.
[3] Burlacu OE, Hajiyan M. Simultaneous localization and mapping literature survey, Advanced Control System ENGG 6580, Acedemia. edu, (2012).
[4] Ho TS, Fai YC, Ming ESL. Simultaneous localization and mapping survey based on filtering techniques. Proceedings of the 2015 tenth Asian control conference (ASCC). IEEE; 2015. p. 1–6.
[5] Cadena C, Carlone L, Carrillo H, Latif Y, Scaramuzza D, Neira J. Past, present, and future of simultaneous localization and mapping: toward the robust-perception age. IEEE Trans Robot 2016;32:1309–32.
[6] Petrovskaya A, Perrollaz M, Oliveira L, Spinello L, Triebel R, Makris A, et al. Awareness of road scene participants for autonomous driving. Handbook of intelligent vehicles. London: Springer; 2012. p. 1383–432.
[7] Wang C-C, Thorpe C, Thrun S, Hebert M, Durrant-Whyte H. Simultaneous localization, mapping and moving object tracking. Int J Robot Res 2007;26:889–916.
[8] Darms M, Rybski P, Urmson C. Classification and tracking of dynamic objects with multiple sensors for autonomous driving in urban environments. Proceedings of the 2008 IEEE intelligent vehicles symposium. IEEE; 2008. p. 1197–202.
[9] Migliore D, Rigamonti R, Marzorati D, Matteucci M, Sorrenti DG. Use a single camera for simultaneous localization and mapping with mobile object tracking in dynamic environments. Proceedings of international workshop on Safe navigation in open and dynamic environments application to autonomous vehicles. 2009.
[10] Petrovskaya A, Thrun S. Model based vehicle detection and tracking for autonomous urban driving. Autonomous Robots 2009;26:123–39.
[11] Vu T-D, Aycard O. Laser-based detection and tracking moving objects using data-driven Markov chain Monte Carlo. Proceedings of the IEEE international conference on robotics and automation, 2009, ICRA'09. IEEE; 2009. p. 3800–6.
[12] Lin K-H, Wang C-C. Stereo-based simultaneous localization, mapping and moving object tracking. Proceedings of the 2010 IEEE/RSJ international conference on intelligent robots and systems (IROS). IEEE; 2010. p. 3975–80.
[13] Vu T-D, Burlet J, Aycard O. Grid-based localization and local mapping with moving object detection and tracking. Inf Fusion 2011;12:58–69.
[14] Azim A, Aycard O. Layer-based supervised classification of moving objects in outdoor dynamic environment using 3D laser scanner. Proceedings of the 2014 IEEE intelligent vehicles symposium. IEEE; 2014. p. 1408–14.
[15] Baig Q, Perrollaz M, Laugier C. A robust motion detection technique for dynamic environment monitoring: a framework for grid-based monitoring of the dynamic environment. IEEE Robot Autom Mag 2014;21:40–8.
[16] Asvadi A, Peixoto P, Nunes U. Detection and tracking of moving objects using 2.5 d motion grids. Proceedings of the 2015 IEEE eighteenth international conference on intelligent transportation systems (ITSC). IEEE; 2015. p. 788–93.
[17] Chang C-H, Wang S-C, Wang C-C. Exploiting moving objects: multi-robot simultaneous localization and tracking. IEEE Trans Autom Sci Eng 2016;13:810–27.
[18] Mertz C, Navarro-Serment LE, MacLachlan R, Rybski P, Steinfeld A, Suppe A, Urmson C, Vandapel N, Hebert M, Thorpe C. Moving object detection with laser scanners. J Field Robot 2013;30:17–43.
[19] Niedfeldt PC. Recursive-RANSAC: a novel algorithm for tracking multiple targets in clutter. Brigham: Young University; 2014.
[20] Qiu C, Zhang Z, Lu H, Luo H. A survey of motion-based multitarget tracking methods. Progress Electromagn Res B 2015;62:195–223.
[21] Niedfeldt PC, Beard RW. Multiple target tracking using recursive RANSAC. Proceedings of the 2014 American control conference. IEEE; 2014. p. 3393–8.
[22] Yang SW, Wang CC. Simultaneous egomotion estimation, segmentation, and moving object detection. J Field Rob 2011;28:565–88.
[23] Raguram R, Frahm J-M, Pollefeys M. A comparative analysis of RANSAC techniques leading to adaptive real-time random sample consensus. Proceedings of the European conference on computer vision. Springer; 2008. p. 500–13.
[24] Niedfeldt PC. Recursive-RANSAC: A novel algorithm for tracking multiple targets in clutter. Brigham Young University; 2014.
[25] Betke M, Wu Z. Data association for multi-object visual tracking. Synthesis Lect Comput Vis 2016;6:1–120.
[26] Guivant JE, Nebot EM. Optimization of the simultaneous localization and map-building algorithm for real-time implementation. IEEE Trans Robot Autom 2001;17:242–57.
[27] Bailey T. Mobile robot localisation and mapping in extensive outdoor environments. The University of Sydney; 2002.
[28] Borges GA, Aldon M-J. Line extraction in 2D range images for mobile robotics. J Intell Robot Syst 2004;40:267–97.
[29] Fischler MA, Bolles RC. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun ACM 1981;24:381–95.
[30] Civera J, Grasa OG, Davison AJ, Montiel J. 1-Point RANSAC for extended Kalman filtering: application to real-time structure from motion and visual odometry. J Field Robot 2010;27:609–31.
[31] Shi J, Li Y, Qi G, Sheng A. Extended target tracking filter with intermittent observations. IET Signal Process 2016;10:592–602.

Masoud S. Bahraini received his B.S. degree from Shahid Bahonar University of Kerman, Kerman, Iran, in 2009; and M.S. degree from Shiraz University, Shiraz, Iran, in 2012, both in Mechanical Engineering. He is currently a Ph.D. candidate in Department of Mechanical Engineering, Yazd University, Yazd, Iran. He is also a research assistant at the Autonomous and Intelligence Systems Laboratory (AISL), Simon Fraser University, Surrey, BC, Canada. His research interests include robot navigation in dynamic environments, deep learning, vibration analysis and control.

Mohammad Bozorg received his B.S. and M.S. and Ph.D. degrees all in Mechanical Engineering from Shahid Chamran University of Ahwaz, Ahwaz, Iran, Sharif University of Technology, Tehran, Iran and Sydney University, Sydney, Australia in 1989, 1991 and 1997, respectively. In 1997, he joined the Department of Mechanical Engineering, Yazd University, Yazd, Iran, where he is currently an associate professor. His current research interests include sensor fusion, navigation of autonomous robots, robust control, control of time-delay systems and mechatronics. He is a member of "Control Design" Technical Committee of International Federation of Automatic Control (IFAC) and a founding member of Robotic Society of Iran (RSI).

Ahmad B. Rad is a Professor with the School of Mechatronic Systems Engineering, Simon Fraser University, Surrey, BC, Canada. His current research interests include autonomous and intelligent systems, robotics navigation, and intelligent control.