Laser-based tracking of multiple interacting pedestrians via on-line learning

Xuan Song,*, Jinshi Cui, Huijing Zhao, Hongbin Zha, Ryosuke Shibasaki

Abstract

Successful multi-target tracking requires locating the targets and labeling their identities. For a laser-based tracking system, this mission becomes significantly challenging when the targets are in close proximity or frequently interact with one another. This paper presents a novel on-line learning-based method for laser-based multi-target tracking. When the targets do not interact with one another, multiple independent trackers are employed to train a classifier for each target. When the targets are in close proximity, the learned classifiers are used to assist in tracking. The tracking and learning supplement each other in the proposed method, which is helpful in dealing with difficult problems encountered in laser-based multi-target tracking; moreover, it ensures that the entire process can be completely automatic and available on-line. Various evaluations have demonstrated that this method performs better than previous methods when interactions occur, and it can maintain the correct tracking under various complex tracking situations.

1. Introduction

Multi-target tracking plays an important role in various applications, such as surveillance and in human motion and traffic flow analyses. Compared to a traditional vision-based tracking system, laser range scanners, as a new type of measurement instrument, have received an increasing amount of attention in recent years for solving different tracking problems. A typical laser-based tracking system [1–3] is illustrated in Fig. 1.

For a laser-based tracking system, multi-target tracking is much easier to achieve when the targets are far apart or not interacting with one another since multiple independent trackers can be used. However, because laser scanners cannot provide color information, when the targets are in close proximity or frequently interact with one another, obtaining their correct trajectories offers some significant challenges. For example, making a correct data association becomes rather difficult. In addition, independent trackers are unavailable owing to the use of merged measurements. Specifically, as illustrated in Fig. 2, maintaining the correct tracking seems to be an impossible task when a well-known "merge/split" condition occurs. Hence, the goals of this research are as follows: First, to devise a new method for helping obtain better tracking performance using laser scanners than in previous methods when interactions occur. Second, to generate a new attempt at solving the "merge/split" problem in laser-based multi-target tracking.

* Corresponding author. Tel.: +81 80 3496 1516.
E-mail address: songxuan@csis.u-tokyo.ac.jp (X. Song).

© 2013 Elsevier B.V. All rights reserved.
depicted through a discriminative model with a supervised
learning process, this model can consider information from
"confusing targets" and can sufficiently exploit the history of
such targets. Through these discriminative models, we can easily
deal with challenging tracking situations. Second, our method can
automatically switch between tracking and learning, which
allows the two phases to supplement each other in one frame-
work, and ensures that the entire processes can be performed
completely on-line with no human interaction. Of course, some
extra computations are needed for the learning phase. However,
the computational complexity of our method increases linearly
with an increase in the number of targets to be tracked. Hence,
our method can be applied to tracking a large number of
interacting targets, which is intractable for methods that use a
joint state space representation.

Hence, the main contributions of this paper can be summar-
ized as follows: First, we developed a unified on-line learning-
based approach for a laser-based tracking problem, which can
easily deal with some of the challenging situations encountered
during tracking. Second, we designed some dedicated features for
laser scanner data for distinguishing between certain interacting
pedestrians.

The remainder of this paper is structured as follows: In the
following section, other work related to this issue is briefly
reviewed. Next, Section 3 provides the overall tracking approach,
including details regarding learning by tracking, and tracking by
learning. The experiments used in this research and their results
are then presented in Section 4. Finally, some concluding remarks
regarding our proposal are given in Section 5.

Please cite this article as: X. Song, et al., Laser-based tracking of multiple interacting pedestrians via on-line learning, Neurocomputing (2013), http://dx.doi.org/10.1016/j.neucom.2013.02.001
2. Related work

Multiple target tracking (MTT) has been studied extensively, and an in-depth review of the literature regarding this issue can be found in a recent survey by Yilmaz et al. [5]. Typically, multi-target tracking can be achieved through a data association [4]. The multi-hypothesis tracker (MHT) [6,7] attempts to keep track of all possible association hypotheses over time, and can be viewed as the most successful algorithm for multi-target tracking based on a data-oriented perspective. However, the MHT algorithm is computationally exponential, both in memory and time, which makes its application difficult. The nearest neighbor standard filter (NNSF) [4] associates each target with the closest measurement in the target state space, and employs a particle filter [8] or Kalman filter [9,10] to complete its tracking. However, this simple procedure prunes away many feasible hypotheses, and cannot solve the “labeling” problems when the targets are crowded together. In this respect, a wider approach to multi-target tracking is achieved by exploiting a joint state space representation that concatenates the states of all the targets together [11–13], or inferring this joint data association problem through the characterization of all possible associations between the targets and observations, such as through a joint probabilistic data association filter (JPDAF) [4,14,15], Monte Carlo technique-based JPDA algorithms (MC-JPDAF) [16,17], and a Markov chain Monte Carlo data association (MCMC-DA) [18–20]. More recently, Huang et al. [21] used the “low-frequency” terms of a Fourier decomposition to represent the distributions over the data association. This method can maintain and update the permutation distribution directly in the Fourier domain, allowing for polynomial time band-limited approximations. However, with an increase in the number of tracking targets, the state space becomes increasingly large, and obtaining an accurate MAP estimation in a large state space becomes quite difficult. Furthermore, the computational complexity of most of the methods mentioned above grows exponentially with an increase in the number of tracking targets.

Additionally, researchers have also proposed the use of multiple parallel filters to track multiple targets [22,23], that is, one filter per target where each target has its own small state space. In spite of this, when interactions among targets occur, this method encounters a difficulty in maintaining the correct tracking. Therefore, modeling the interactions among the targets becomes an incredibly important issue. Khan et al. [24] used a Markov random field (MRF) motion prior to modeling the interactions among targets. Qu et al. [25] proposed the use of a magnetic-inertia potential model to handle the “merge error” problem. Yu et al. [26] allowed for collaboration among filters by modeling the joint distribution of the targets prior to using a Markov random network to solve the “merge error” problem. Lanz et al. [27] proposed a hybrid joint-separable model to deal with interactions among targets. Sullivan et al. [28] track both isolated targets and “merging targets,” and then connect these trajectories using a clustering procedure. Nillius et al. [29] employ a track graph to describe when the targets are isolated and how they interact. They utilize a Bayesian network to associate the identities of the isolated tracks by exploiting this graph. However, most of these methods mentioned above (except [28,29]) consider tracking as a Markov process, which fails to sufficiently exploit the target history.

There has been a recent trend toward introducing learning techniques into object target tracking or range data detection. Representative publications include [30,31,34]. The authors in [30,31,34] formulate a tracking task as a classification problem and continuously update the current boosting classifier that represents an object to optimally discriminate the object from the current background. Lu et al. [33] utilized random image patches to represent the foreground of an image and extract it from the background using on-line updating. Li et al. [32] proposed a particle-filter based tracker using different on-line learned observers. Arras et al. [35] utilized geometrical features to train the boosting classifier to detect individuals from the laser data. The authors of [43,36] utilized some on-line trained classifiers to describe the appearance model of the targets, and attempted to deal with interactions and occlusions among targets. Babu et al. [38,39] developed a novel on-line adaptive object tracker based on fast learning radial basis function (RBF) networks, and trained two separate RBF networks for object and non-object pixels. Suresh et al. [40] presented an on-line learning neural tracker (OLNT) to differentiate an object from its background, and to adapt to changes in the object/background dynamics. Williams et al. [41] utilized a probabilistic relevance vector machine (RVM) to generate the observations of objects and build a displacement expert that can directly estimate the displacement from the target region to ensure robust tracking. Finally, Song et al. [42,44] developed a fusion system that combines camera and laser-scanner sensors to perform joint tracking using an on-line learning technique. Because such a kind of system needs both laser-scanners and cameras, it usually needs more engineering cost for application.

However, most of the researches described above focus mainly on how to classify a single target from its background in the visual data, or how to perform multi-target tracking using a camera sensor. In contrast, the laser-based multi-target tracking problem is quite different owing to the different measurements achieved from the two sensors, and its advantages are also obvious: The computational costs of generating laser-scanner data are much smaller than the visual data. In addition, it can easily obtain a robust detection in a real environment [44] because it is insensitive to changes in lighting and weather conditions. Therefore, in this paper, we transfer some exciting concepts of an on-line learning technique into a laser-based multi-target tracking problem and propose a unified framework to couple the learning and tracking. During this procedure, we have solved two crucial problems: learning by tracking, in which we solve the problem of how to obtain positive and negative samples without human interaction to achieve a completely on-line supervised learning; and tracking by learning, in which we solve the problem of how to use these learned classifiers to deal with difficult problems (interaction or merge/split) encountered in multi-target tracking.
3. Tracking approach

The overall tracking framework is illustrated in Fig. 5. It mainly contains two processes: learning by tracking (green module in Fig. 5) and tracking by learning (blue module). Before that, the tracking situation detection module makes the system switch automatically between independent tracking and interacting targets tracking (yellow module). (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

3.1. Learning by tracking

When the targets are far apart or do not interact with one another (non-correlated targets), the tracking becomes relatively easy since it can be solved through multiple independent trackers. Specifically, the results obtained are sometimes accurate and credible. Consequently, these tracking results can be utilized as samples for supervised learning. Hence, we call this process “learning by tracking.” In this section, we provide details on independent tracking, feature extraction, and our on-line learning algorithm.

3.1.1. Independent tracking

We can formulate the sequential tracking problem by computing the maximum a posteriori (MAP) $\mathbf{x}^*$, such that $\mathbf{x}^* = \arg \max p(\mathbf{x} | \mathbf{z})$, where $\mathbf{x} = (x_0, \ldots, x_t)$ is a state sequence, and $\mathbf{z} = (z_0, \ldots, z_t)$ is an observation sequence. A basic Bayesian sequential estimation can be described as a two-step recursion.

Prediction step:

$$ p(x_t | z_{t-1}) = \int p(x_t | x_{t-1}) p(x_{t-1} | z_{t-1}) dx_{t-1}. $$

(1)

Filtering step:

$$ p(x_t | z_t) = \frac{p(z_t | x_t) p(x_t | z_{t-1})}{p(z_t | z_{t-1})}, $$

(2)

where $p(x_t | x_{t-1})$ is the transition distribution (dynamic model), and $p(z_t | x_t)$ is the likelihood (observation model).

Clearly, we can utilize a Kalman or particle filter to compute Eqs. (1) and (2). In this research, we chose the use of a particle filter as an independent tracker for the following reasons: First, a human walking model is sometimes very complicated, and it is difficult to be described using a linear transition function. Therefore, a Kalman filter will encounter difficulties when dealing with this issue. In addition, the observation model sometimes contains multiple components, the weights of which should be automatically adjusted (e.g., interacting-target tracking situations). In this case, a particle filter will be more convenient for computing a MAP estimation. Finally, as shown in the discussions in [23], a particle filter can easily maintain correct tracking compared with a Kalman filter when occlusions occur.

The basic idea behind a particle filter is very simple. Starting with a weighted set of samples $[w_t^n, x_t^n]$, approximately distributed according to $p(x_t | z_{t-1})$, new samples are generated from a suitably designed proposal distribution $q(x_t | x_{t-1}, z_t)$. To maintain a consistent sample, the new importance weights are set to

$$ w_{t+1}^n \propto w_{t}^n \frac{p(z_t | x_t^n) p(x_t^n | z_{t-1})}{q(x_t^n | x_{t-1}^n, z_t)}. $$

(3)

More details on the particle filter technique can be found in [8]. In the following, we provide details on these models in our particle-based independent tracker.

State space and dynamic model: The state of an individual at time $t$ is modeled using an eight-dimensional state vector, $\mathbf{x}_k = \begin{bmatrix} p_{k}^L, p_{k}^R, T_{k}, S_{k}, a_{k}^L, a_{k}^R \end{bmatrix}$, where $k$ is the individual’s identifier, $p_{k}^L$ and $p_{k}^R$ denote the positions of the individual's feet, which make up a two-dimensional vector, $\mathbf{p}_k = (x, y)$. $T_k$ and $S_k$ are the walking period and walking stride of an individual, respectively. Finally, $a_{k}^L$ and $a_{k}^R$ are the acceleration of the individual’s feet. In this research, we do not add the velocity into the state space, as this will complicate the overall inference process and decrease the accuracy of our approach (particularly during the sampling and updating processes). However, we can utilize a second-order autoregressive dynamical function to perform a prediction, which contains the velocity information.

Therefore, a typical walking model, as illustrated in Fig. 6, is utilized and is as follows:

If the left foot swings:

if the left foot is the rear foot $a_{k}^L = a$, $a_{k}^R = 0$
else $a_{k}^L = -a$, $a_{k}^R = 0$

$$ p_{k}^L = p_{k-1}^L + \Delta T p_{k-2}^L + C a_{k}^L + N_{k}^L, $$

(4)

$$ p_{k}^R = p_{k-1}^R + N_{k}^R, $$

(5)

$$ T_{k} = T_{k-1} + N_{T}, $$

(6)
where matrices $A$, $B$, and $C$ are constant ratios obtained through a regression using 150 representative sequences. This function contains the velocity information, and is better than a simple constant velocity or acceleration motion function. $N_{p}$, $N_{y}$, $N_{f}$, and $N_{S}$ are normally distributed random noise. In this research, we utilized this dynamic model as the proposal distribution, $q(x_{i}|x_{t-1},z_{i})$, for generating new samples in the particle filter.

**Observation model:** The observation is a point set in this short-term period. This information can also be used for learning. Hence, a walking pattern similarity can be described as follows:

$$S_{d}(t) = -\frac{1}{48} A T^{2} \cos \left(2 \pi \frac{t}{T} \right) = -S \cos(\gamma),$$

where $A$ stands for the maximum acceleration, $T$ indicates the walking period, $S$ represents the stride, and $\gamma$ is the walking phase. Hence, a walking pattern similarity can be described as follows:

$$P_{\text{walk}}(z_{i}|x_{k,t}) = \frac{1}{\sqrt{2\pi\delta_{w}}} \exp \left(-\frac{(S_{d} - S_{d,t}\cos(2\pi t/T_{k,t}))^2}{2\delta_{w}^2} \right).$$

where $S_{d}$ is the distance between two feet.

We assume that these similarities are independent from each other within a short interval, an assumption that has been demonstrated as workable in our previous research \[23,45\]. Hence, the likelihood function used for updating the filter is as follows:

$$P(z_{i}|x_{k,t}) = P_{\text{pos}}(z_{i}|x_{k,t}) \times P_{\text{walk}}(z_{i}|x_{k,t}) \times P_{\text{pred}}(z_{i}|x_{k,t}).$$

Hence, after re-weighting and re-sampling during the particle filter process, we can obtain the new position of each pedestrian.

### 3.1.2. Discriminative features for learning

Once we obtain the tracking results for each target, to perform on-line supervised learning, we should extract some discriminative features and save them in a sample pool. In this subsection, we present some discriminative features for such learning, which can be divided into motion features and walking pattern features. We believe that different individuals have different walking habits. Moreover, different individuals should have specific motion information (velocity, acceleration, etc.) over a short-term period. This information can also be used for learning.

The training samples were given by a set of segments together with their labels

$$E = \{\text{Seg}_{i}, f_{i}\} | i \in \{+1,-1\},$$

where segment $\text{Seg}_{i}$ is the detection or tracking result representing a pedestrian with two feet. In addition, $f_{i} \in \{+1,-1\}$, where $f_{i} = +1$ denotes that this segment is a positive sample, and $f_{i} = -1$ denotes that is a negative one. We define the feature $f$ as a function $f: \text{Seg} \rightarrow \mathbb{R}$, which takes a segment $\text{Seg}$ as an argument and returns a real value. For each segment $\text{Seg}_{i}$, we can determine the following feature extraction function:

$$f(S, T).$$

Eqs. (4) and (8) are second-order autoregressive functions, where matrices $A$, $B$, and $C$ are constant ratios obtained through a regression using 150 representative sequences. This function contains the velocity information, and is better than a simple constant velocity or acceleration motion function. $N_{p}$, $N_{y}$, $N_{f}$, and $N_{S}$ are normally distributed random noise. In this research, we utilized this dynamic model as the proposal distribution, $q(x_{i}|x_{t-1},z_{i})$, for generating new samples in the particle filter.

**Observation model:** The observation is a point set in this short-term period. This information can also be used for learning. Hence, a walking pattern similarity can be described as follows:

$$S_{d}(t) = -\frac{1}{48} A T^{2} \cos \left(2 \pi \frac{t}{T} \right) = -S \cos(\gamma),$$

where $A$ stands for the maximum acceleration, $T$ indicates the walking period, $S$ represents the stride, and $\gamma$ is the walking phase. Hence, a walking pattern similarity can be described as follows:

$$P_{\text{walk}}(z_{i}|x_{k,t}) = \frac{1}{\sqrt{2\pi\delta_{w}}} \exp \left(-\frac{(S_{d} - S_{d,t}\cos(2\pi t/T_{k,t}))^2}{2\delta_{w}^2} \right).$$

where $S_{d}$ is the distance between two feet.

We assume that these similarities are independent from each other within a short interval, an assumption that has been demonstrated as workable in our previous research \[23,45\]. Hence, the likelihood function used for updating the filter is as follows:

$$P(z_{i}|x_{k,t}) = P_{\text{pos}}(z_{i}|x_{k,t}) \times P_{\text{walk}}(z_{i}|x_{k,t}) \times P_{\text{pred}}(z_{i}|x_{k,t}).$$

Hence, after re-weighting and re-sampling during the particle filter process, we can obtain the new position of each pedestrian.

### 3.1.2. Discriminative features for learning

Once we obtain the tracking results for each target, to perform on-line supervised learning, we should extract some discriminative features and save them in a sample pool. In this subsection, we present some discriminative features for such learning, which can be divided into motion features and walking pattern features. We believe that different individuals have different walking habits. Moreover, different individuals should have specific motion information (velocity, acceleration, etc.) over a short-term period. This information can also be used for learning.

The training samples were given by a set of segments together with their labels

$$E = \{\text{Seg}_{i}, f_{i}\} | i \in \{+1,-1\},$$

where segment $\text{Seg}_{i}$ is the detection or tracking result representing a pedestrian with two feet. In addition, $f_{i} \in \{+1,-1\}$, where $f_{i} = +1$ denotes that this segment is a positive sample, and $f_{i} = -1$ denotes that is a negative one. We define the feature $f$ as a function $f: \text{Seg} \rightarrow \mathbb{R}$, which takes a segment $\text{Seg}$ as an argument and returns a real value. For each segment $\text{Seg}_{i}$, we can determine the following feature extraction function:

$$f(S, T).$$
Motion features: (1) Foot velocity: This feature measures the mean speed of a specific foot in segment i, which is given by
\[ v^i = \sqrt{(x^i_t - x^i_{t-1})^2 + (y^i_t - y^i_{t-1})^2} / \Delta t, \]  
where \((x^i_t, y^i_t)\) is the barycenter of one foot in segment i, and \(\Delta t\) is the interval.

(2) Foot velocity direction: This feature measures the velocity direction for a specific foot in segment i, which is given by
\[ \alpha^i = \arctan(v_{y^i_t} / v_{x^i_t}), \]  
where \(v_{x^i_t}\) and \(v_{y^i_t}\) are the x-coordinate and y-coordinate velocities.

(3) Foot acceleration: This feature measures the mean acceleration for a specific foot in segment i, which is given by
\[ a^i = \sqrt{(v_{x^i_{t+1}} - v_{x^i_{t-1}})^2 + (v_{y^i_{t+1}} - v_{y^i_{t-1}})^2} / \Delta t. \]  
(4) Foot acceleration direction: This feature measures the acceleration direction for a specific foot in segment i, which is given by
\[ \alpha^i_a = \arctan(v_{y_{t+1}} - v_{y_{t-1}}) / (v_{x_{t+1}} - v_{x_{t-1}}). \]

Therefore, we can obtain eight values to depict the above features from one segment.

Walking pattern features: When a pedestrian takes a step forward, as a typical motion, one foot may swing as the other foot pivots. The two feet interchange their duties by landing and moving shifts with a rhythmic pattern. If we place all the laser points into the spatial-temporal domain, it can be seen that some periodic braided-like movements are generated, as shown in Fig. 7.

As can be seen from the above discussion, the distance between two feet, \(S_6\), varies in a cosine pattern, which can be described by Eq. (14). Obviously, stride \(S\) and walking period \(T\) can be used as discriminative features for this type of learning. When the targets are far apart, extracting the two features is very easy, as the two parameters are in a state that makes them easy to estimate. However, for correlated targets, a difficulty may arise since we cannot complete an updated procedure for estimating the state. Hence, we introduce another strategy to extract the two features under this condition. For \(n\) continued frames, we obtain the values for \(S_j\) and \(T\) (we assume they are in the same period). Therefore, extracting features \(S\) and \(T\) becomes a parameter estimation problem:

\[ S_j = -S \cos \left(2\pi \frac{v_j}{T} \right), \quad j = 1, \ldots, n. \]

![Fig. 7. An S-T representation of feet data: There were three pedestrians in this figure. Obviously, the walking patterns of them generated in different styles.](image)
for updating a classifier: once we initialize a classifier, it is placed into service for a specific time, and samples are collected in a sample pool for the next synchronous update. The entire learning algorithm is shown below.

Learning Algorithm

**Input:** Feature vectors of targets and their labels \([s^t_{k',t}, f^t_{k',t}]\), \(t = 1, \ldots, T\)

**Output:** The strong classifier \(H(s_{k,t})\) of target \(k\) at time \(t\)

### Train a Strong Classifier (for time \(t\))
1. Initialize weights \(w_i^0\) to be \(1/N\).
2. For \(j = 1 \ldots M\) (train \(M\) weak classifiers)
   a. Make \(w_i^j\) a distribution.
   b. Train a weak classifier \(h_j\).
   c. Set \(err_j = \sum_{s_i} w_i |h_j(s_i) - f^t_{k',t}|\).
   d. Update weak classifier weight \(z_j = 0.5 \log(1-err_j/err)\).
   e. Update example weights \(w_i = w_i e^{-h_j(s_i) - z_j}\).
3. The strong classifier is given by \(H(s_{k,t})\), where \(H(s_{k,t}) = \sum_j z_j h_j(s_{k,t})\).

### Update the Strong Classifier (for time \(t\))
1. Initialize weights \(w_i^0\) to be \(1/N\).
2. For \(j = 1 \ldots K\) (choose \(K\) best weak classifiers and update their weights)
   a. Make \(w_i^K\) a distribution.
   b. Choose \(h_j(s_{k,t-1})\) with minimal \(err_j\) from \(h_j(s_{k,t-1})\).
   c. Update \(z_j\) and \(w_i^K\).
   d. Remove \(h_j(s_{k,t-1})\) from \(h_j(s_{k,t-1})\).
3. For \(j = K+1, \ldots, M\) (add new weak classifiers)
   a. Make \(w_i^K\) a distribution.
   b. Train a weak classifier \(h_j\).
   c. Compute \(err_j\) and \(z_j\).
   d. Update example weights \(w_i^K\).
4. The updated strong classifiers are given by \(H(s_{k,t})\), where \(H(s_{k,t}) = \sum_j z_j h_j(s_{k,t})\).

### Tracking by learning

When the targets are in close proximity, it is difficult to maintain the correct tracking using independent trackers. Specifically, when “merge/split” conditions occur, associating the identities of the targets becomes a significantly challenging problem. In this case, the learned classifiers can be utilized to assist in the tracking. In this section, we provide details on how to employ these classifiers to deal with the difficult problems encountered during tracking.

#### 3.2.1. Correlated targets tracking

As described in [44], once a target finds more than one detection and is not a “merged target,” we conclude that there some correlated targets exist. In this case, an independent tracker can no longer provide reliable results, and the observation model for a specific target contains multiple components whose weights should be adjusted. Hence, learned classifiers should be used; we use these detection candidates and the previous state of this target to extract the features depicted in Section 3.1.2. The obtained feature vectors of these detection candidates are input into the classifier, \(H(s_{k,t})\), and we utilize the classification margin as the output of these classifiers. Therefore, the output of these classifiers is certain scores that can be utilized to measure the chances of a particular detection candidate associating which a particular target. We normalize the scores from these detection candidates, between 0 and 1, and make them sum up to 1. Therefore, it can be used as the weight of this detection during an observation. In this case, the observation for target \(k\) can be computed as

\[
P(z_t | x_{k,t}) = \sum_{u=1}^{N} \beta_{u,k} P_{u,k}(z_u | x_{k,t}),
\]

where \(P_{u,k}(z_u | x_{k,t})\) is the observation of detection candidates \(u\) computed by (16), \(\beta_{u,k}\) the weights of detection candidates \(u\) depending on the classifiers, and \(J\) is the number of detection candidates.

Eq. (25) provides several advantages when we deal with correlated-target tracking: First, since the correlated targets are collected as positive or negative samples of each other during the learning process, the classifier for each target becomes a discriminative model that can usually make \(\beta_{u,k}\) differ greatly. Hence, it can not only consider the information from other targets, but can also easily discriminate each other. In addition, since an observation contains multiple components, even when the weights of the different segments are only approximations (sometimes the distinguishing ability of the classifiers is not sufficiently strong), such an observation ensures that, at worst, imprecise results are obtained, rather than a completely incorrect data association.

An example is shown in Fig. 9. In frame 560, targets 4 and 5 are in close proximity. In this case, the observation of each target contains two segments, the weights of which are dependent on the classifier. After the classifier weighting, we can see that, for a specific target, the observation of a target position is more peaked around the actual position. As a result, the particles are more focused around the true target state after the re-weighting and resampling of each level using the filter process, and we can easily obtain the correct tracking.

#### 3.2.2. Merge/split condition

It is sometimes difficult to obtain good separate detections, or associate different legs with a specific person when several individuals are walking together. Maintaining the correct tracking and extracting features becomes impossible under such a condition. Hence, we deal with this as a merge/split condition.

Once we detect that some targets are merging, as explained in [44], we assign a new state space \(x^n_{k,t} = [P^n_{k,t}, \phi]\) for this “merged target,” where \(P^n_{k,t} = [X,Y]\) is the barycenter position of the “merging segment,” and \(\phi\) indicates which targets are merging. During the filtering process, we do not update parameter \(\phi\). We employ a particle filter-based tracker to track the “merged target,” which is similar to the tracker described in Section 3.1.1. However, the dynamic model and the observation model should be as follows:

\[
P^n_{k,t} = A^n P^n_{k,t-1} + \Xi^n P^n_{k,t-2} + N^n,
\]

\[
P_{\text{merge}}(z_t | x_{k,t}) = \frac{1}{\sqrt{2 \pi \sigma_p}} \times \exp \sum_{i=1}^{N} \left( -\frac{(x^n_i - x^n_t)^2 + (y^n_i - y^n_t)^2}{2 \sigma_p^2} \right).
\]

Once we detect that the “merged target” has split, we should re-initialize the state of the targets and track them using independent trackers. Their discriminate features are then extracted...
and input into their respective classifiers. We normalize the scores obtained by these classifiers and specify the correct links for their trajectories. An example of this process is shown in Fig. 10(a)–(e). In contrast, the results from a constant velocity tracker are shown in Fig. 10(f)–(h). The state space of this tracker is $x_{k,t} = [x, y, v_x, v_y]$, where $(x, y)$ is the position of the target, and $(v_x, v_y)$ is the velocity. We utilize the constant velocity function as the motion model (without using a “two feet walking model”),
and the observation model is similar to that described through Eq. (27). From these results, we can see that the trajectories of these trackers are not smooth from this simple motion model (containing incorrect localizations and noise), and the IDs of targets 2 and 3 were switched in the final results.

Therefore, with the help of the classifiers, our method can deal with various complex tracking situations. In addition, the tracking and learning supplement each other in the proposed method, consequently becoming an adaptive loop, which ensures that the entire process can be completely utilized on-line.

4. Experiments and results

We evaluated the proposed method in a real plaza (an area of about 30 m x 25 m), as well as in a park and a crossing area. We performed two groups of experiments. In the first group of experiments, a single-row laser scanner (LMS291) produced by SICK was utilized. The scanner was set on the ground and performed horizontal scanning at a frequency of 37 fps. In the second set of experiments, we utilized three laser scanners to deal with occlusions. In addition, we added a camera to present the results more intuitively. We utilized a time server to deal with the time synchronization problem between the different sensors. The extrinsic calibration was conducted using several control points in a box. For more details regarding this aspect of the experiments, refer to [45]. The selected data used for testing were different clips consisting of more than 57,000 frames, in which complex interactions frequently took place.

All the algorithms mentioned were implemented in the experiments using a non-optimized MATLAB code with C++ interfaces. All the simulations were run on a PC with an Intel 1.6 GHz dual core processor. The results, comparisons, and specific parameter choices are detailed in the following subsection.

4.1. Behaviors of the classifiers

In this subsection, we present some experimental results to show the behaviors of the classifiers during the tracking process.

First, we should determine the number of weak classifiers M for on-line learning. Therefore, we utilized independent trackers (without learning) to track 5000 frames, and obtained about 23 trajectories to perform a cross-validation. During this process, we did not add or remove the weak classifiers, and the overall process was off-line learning. We set the number of weak classifiers to 1–10, and the resulting classification accuracy is shown in Fig. 11. From this experiment, we found that when the number of weak classifiers reached eight, the growth rate of the classification accuracy became very slow. Considering the computational complexity and overfitting problem, we set M = 8 in the experiments described below. In addition, from this figure, we can see that when five weak classifiers were used, the classifiers could maintain most of the history information of the targets. Hence, for on-line learning, we kept five weak classifiers and dropped three, one during each update.

In addition, during the on-line learning process, we found that the performance of a classifier was sometimes related to its life
span: If the life-span was too short, we did not have enough samples to train a strong classifier. If the life-span was too long, the individuals might change their walking styles, which prevented the classifiers from updating in time. The time cost expended for learning is also related to this parameter. We set the classifier life span at 30, 60, 90, 120, 150, 180, and 210 frames, respectively, and took a statistical survey of 5000 frames. In these frames, we extracted the features for one sample per five frames, and trained eight weak classifiers for each target. Further details of this experiment are shown in Fig. 12. From this figure, we can see that the correct rate of the classifier reached its maximum (78.53%) when its life span was 120, which provided the best trade-off between the performance of the classifier and the computational cost of the algorithm. As described in the section below, we performed all the experimental evaluations using these parameters.

Fig. 13 shows the behaviors of the weak classifiers, which reflect the impact of the different features during the tracking process. The top row shows the tracking sequences, while the bottom row shows the behavior of the selected weak classifier. Each bin indicates a feature, and the order of features is the same as described in Section 3.1.2. The magnitude of the bars indicates the weight of the feature. Through this experiment, we found that, in most cases, an individual’s walking stride, walking period, and velocity direction of their feet are three important discriminative features for learning and tracking.

4.2. Tracking results

4.2.1. Single laser scanner

Fig. 14 shows some selected tracking results under interacting situations using a single laser scanner, where the blue points are the raw laser data, the red points denote the obtained feet positions, and the circles indicate the target position. In Fig. 14, six pedestrians were walking together. In frame 561, targets 4 and 5 began merging together. However, 149 frames later, we still maintained the correct trajectories for both targets.

Fig. 13. Behavior of classifiers: The top row shows the tracking sequences, while the bottom row shows the behavior of selected weak classifier. Each bin indicates a feature and the order is the same as the description in Section 5, which is the velocity of feet, velocity direction of feet, acceleration of feet, acceleration direction of feet, walking stride and walking period, respectively. The magnitude of the bars indicates the weight of the feature. (a) As can be seen, in frame 6300, the velocity direction of feet, the walking stride and walking period played an important role in the learning and testing. But in frame 6500, the walking stride and walking period played a more important role. (b) In frame 113, the velocity direction of feet was the very important discriminative feature. But in frame 297, the walking stride and walking period played a more important role in the learning and testing.

Fig. 14. Tracking results in the plaza with single laser scanner: Note that targets 4 and 5 were merging in frame 561, and the final trajectories of these targets were shown in the last sub-figure. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)
We conducted a statistical survey of 6000 continuous frames and counted the challenging situations (such as merges/splits, correlated targets, and occlusions) occurring in these frames to show how many complex situations our method can deal with. When one of these conditions occurred without causing a tracking failure, it was counted as a successful disposal. Further details on this are shown in Table 1.

From this table, we can see that our method can deal with most correlated targets and merge/split conditions. However, we found that the proposed method has difficulties in obtaining good results once an occlusion or an uncertain detection occurs: For occlusions, once consecutive occlusions occur, our method has a difficulty in estimating the states of the targets and judging the tracking situations, which sometimes causes consecutive tracking failures. Uncertain detections, such as the non-detection of targets, an incorrect clustering of segments, and false alarms, also have a negative influence on our method. These uncertain detections may cause an incorrect initiation of targets, the false appearance/disappearance of the targets, and so on. For instance, if a group of targets appear in a frame together, it is difficult to cluster them into different separated targets. Our method has difficulty dealing with such a problem since the initialization of the targets is incorrect.

### 4.2.2. Multiple laser scanners

To deal with the occlusion problem and obtain more robust detections, we utilized multiple laser scanners and performed another group of experiments. In these experiments, three laser scanners and one camera were utilized. The camera was set on the third floor of a building, and was used to present the results more intuitively.

**Fig. 15.** Tracking results in the campus with multiple laser scanners: The first row is the results without on-line learning algorithm, and the second row is the results of our method. Note that the spotlight-rectangle A, B, and C. The trackers without on-line learning algorithm were difficult to maintain the correct tracking when the targets were in close proximity. In contrast, our method could easily deal with these challenging situations and maintain the correct tracking. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

### 4.3. Comparison

**Fig. 16.** Quantitative comparison under interacting situations among three methods: This comparison was conduct among the methods that can track a variable number of targets. (a) shows the correct tracking among three methods in 5000 frames. (b) shows the target number and interactions in 5000 frames.
the targets were in close proximity (as shown from viewpoint A in frame 13,010, viewpoint B in frame 13,036, and viewpoint C in frame 13,104). In contrast, our method could easily maintain the correct tracking even when these challenging situations occurred. In addition, after using the multiple laser scanners, the occlusions and uncertain detections rarely appeared, and the tracking results were significantly improved.

4.3. Quantitative comparison

4.3.1. Tracking a variable number of targets

First, we conducted a quantitative comparison of three methods: an independent particle-filter-based tracker [23] (without on-line learning), an independent Kalman-filter-based tracker [1] (without on-line learning) and ours. We utilized the same state vector and observation model as described in Section 3.1.1 for each of these methods. On the other hand, since all these methods can track a variable number of targets, we performed the evaluations for 5000 continuous frames. The data used for testing were obtained from multiple single-laser scanners. We determined that, if the distance between two persons was less than 30 cm, one interaction would be counted. The ground truth was obtained in a semi-automatic way (trackers + manual labeling). By comparing the tracking results with the ground truth, it was easy for us to recognize different failed tracking situations, including a missed target, a false location, and an identity switch.

Further details regarding this comparison are illustrated in Fig. 16. From this figure, we can see that our method can obtain more robust tracking results than an independent-PF when interactions occur (as shown in frame 1386). On the other hand, the independent-KF had difficulty dealing with some complex situations of multiple walking pedestrians, and tracking failures therefore frequently took place (as shown in frame 3883, for example). In contrast, our method can easily maintain the correct tracking during these challenging situations. The overall success rates of these methods are shown in Table 2. Besides, there are also some new developed learning algorithms, such as [48,49]. However, at present, they are a bit difficult to be applied to our system. In the future, we will try improve them, and perform some experimental comparisons and evaluations.

4.3.2. Tracking a fixed number of targets

Second, we conducted another group of quantitative comparisons of some classical data association methods: a Joint Probabilistic Data Association Filter (JPDAF), a Monte Carlo Joint Probabilistic Data Association Filter (MC-JPDAF), a Nearest Neighbor Standard Filter (NNSF), and our proposed filter. We utilized the same state vector and observation model for each of these algorithms. We utilized the method in [50] to implement the JPDAF algorithm. For the MC-JPDAF, we replaced the filter process with Monte Carlo sampling and updating. Because JPDAF can only track a fixed number of targets, we selected 3000 continuous frames in multiple laser scanner data, and initialized all trackers in frames 1524, 1037, 1631, 1940, and, 2538. The tracking target number and interactions in these frames are shown in Fig. 17(b), and the correct tracking of these methods is shown in Fig. 17(a). From this evaluation, we found that our method can obtain better tracking results than these classical data association methods during interactions, as the proposed method can sufficiently exploit a target’s history, and the on-line trained classifiers can provide stronger distinguishability than these data association methods. The overall success rates of the four methods are shown in Table 3.

In addition, we recorded the average time cost per frame for the four methods to evaluate their computational complexity, the testing data of which was obtained using single laser scanners. Fig. 18 shows the time costs of the four methods for tracking different numbers of targets. We can see that with an increasing number of tracking targets, the time cost of JPDAF and MC-JPDAF grow exponentially as they compute in a joint state space. Compared to these two methods, our method has a linear complexity. When the number of tracking targets reached eight, the time cost of our method was close to the JPDAF, and had a

<table>
<thead>
<tr>
<th>Table 2</th>
<th>Success rate among three methods.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Algorithm</td>
<td>Success rate (%)</td>
</tr>
<tr>
<td>Independent KF</td>
<td>64.28</td>
</tr>
<tr>
<td>Independent PF</td>
<td>76.37</td>
</tr>
<tr>
<td>Ours</td>
<td>85.32</td>
</tr>
</tbody>
</table>

![Fig. 17](http://dx.doi.org/10.1016/j.neucom.2013.02.001) Quantitative comparison under interacting situations among four methods: This comparison was conduct among the methods that can track a fixed number of targets. (a) shows the correct tracking among three methods in 3000 frames. (b) shows the number of tracking targets and the interactions in 3000 frames.
lower time cost than MC-JPDAF. Because all the algorithms were implemented using non-optimized MATLAB code, we believe that our algorithm will have a lower time cost if it is written in C++, which can be used for real-time applications.

5. Conclusion and discussion

In this paper, a novel on-line learning based method was presented for laser-based tracking in wide open areas. Different evaluations indicate the superior tracking performance of the proposed method under certain complex situations. Although the on-line learning requires some extra computations, the computational complexity of our method increases linearly with the number of tracking targets, which will allow it to track a large number of interacting targets.

However, our method does have some limitations. The proposed method sometimes has a difficulty remaining robust once some uncertain detections occur, such as the non-detection of the targets, an incorrect clustering of the segments, or a false alarm. In the future, more robust detection and clustering algorithms should be explored. At the same time, an effective strategy to deal with an uncertain detection should be another focus point of our future work.

Acknowledgments

This work was supported by a Grant-in-Aid for Young Scientists (23700192), GRENE (Environmental Information) Project, and Strategic Project to Support the Formation of Research Bases at Private Universities: Matching Fund Subsidy by Japan Ministry of Education, Culture, Sports, Science, and Technology (MEST). This work was also partially funded by NSFC Grants (no. 60975061) of China. We thank all anonymous reviewers for their helpful comments on this work.

Table 3

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Success rate (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>NSF</td>
<td>63.26</td>
</tr>
<tr>
<td>JPDAF</td>
<td>71.83</td>
</tr>
<tr>
<td>MC-JPDAF</td>
<td>81.71</td>
</tr>
<tr>
<td>Ours</td>
<td>89.35</td>
</tr>
</tbody>
</table>

Fig. 18. Time cost comparison among the four methods: The vertical tick marks are the average time cost per frame, and horizontal tick marks show the number of tracking targets.

References

Xuan Song received his B.S. degree in Information Engineering from Jilin University, China, in 2005 and Ph.D. degree in Signal and Information Processing from Peking University, China, in 2010. From 2010 to 2012, he joined Center for Spatial Information Science, The University of Tokyo as a Post-Doctoral Researcher. In 2012, he was promoted to Project Assistant Professor with the Center for Spatial Information Science, The University of Tokyo. His research interests include computer vision, digital geometry processing, and robotics.

Jinshi Cui received the B.S. and Ph.D. degrees in Computer Science from Tsinghua University, Beijing, China, in 1999 and 2004, respectively. From 2004, she joined the School of Electronics Engineering and Computer Science, Peking University, Beijing, as an Assistant Professor. She was promoted to Associate Professor in 2007. Her research interests include computer vision and robotics.

Huijing Zhao received her B.S. degree in Computer Science from Peking University, Beijing, China, in 1991 and the M.E. and Ph.D. degrees in Civil Engineering from the University of Tokyo, Tokyo, Japan, in 1996 and 1999, respectively. From 1991 to 1994, she was with Peking University, where she was involved with a project developing a graphic information system platform. In 2003, after several years of post-doctoral research with the University of Tokyo, she became a Visiting Associate Professor with the Center for Spatial Information Science. In 2007, she joined Peking University as a Professor with the Key Laboratory of Machine Perception (MOE), and the School of Electronics Engineering and Computer Science. Her research interests include machine perception, intelligent vehicles, and spatial data handling.

Hongbin Zha received the B.E. degree in Electrical Engineering from Hefei University of Technology, Hefei, China, in 1983 and the M.S. and Ph.D. degrees in Electrical Engineering from Kyushu University, Fukuoka, Japan, in 1987 and 1990, respectively. After being a Research Associate with Kyushu Institute of Technology, he joined Kyushu University in 1991 as an Associate Professor. In 1999, he was also a Visiting Professor with the Centre for Vision, Speech, and Signal Processing, Surrey University, Surrey, UK. Since 2000, he has been a Professor at the Center for Information Science, School of Electronics Engineering and Computer Science, Peking University, Beijing, China. He has published more than 200 technical publications in journals, books, and conference proceedings. His research interests include computer vision, digital geometry processing, and robotics.

Ryosuke Shibasaki was born in Fukuoka, Japan. He received his B.S., M.S., and Doctoral degrees in Civil Engineering from the University of Tokyo, Tokyo, Japan, in 1980, 1982, and 1987, respectively. From 1982 to 1988, he was with the Public Works Research Institute, Ministry of Construction. From 1988 to 1991, he was an Associate Professor in the Civil Engineering Department, University of Tokyo. In 1991, he joined the Institute of Industrial Science, University of Tokyo. In 1998, he was promoted to a Professor in the Center for Spatial Information Science, University of Tokyo. His research interests cover three-dimensional data acquisition for GIS, conceptual modeling for spatial objects, and agent-based microsimulation in a GIS environment.