Online Learning for Human Classification in 3D LiDAR-based Tracking

Zhi Yan, Tom Duckett, Nicola Bellotto
{zyan, tduckett, nbellotto}@lincoln.ac.uk
Lincoln Centre for Autonomous Systems, University of Lincoln, UK
http://lcas.lincoln.ac.uk

***Abstract*— Human detection and tracking are essential aspects to be considered in service robotics, as the robot often shares its workspace and interacts closely with humans. This paper presents an online learning framework for human classification in 3D LiDAR scans, taking advantage of robust multi-target tracking to avoid the need for data annotation by a human expert. The system learns iteratively by retraining a classifier online with the samples collected by the robot over time. A novel aspect of our approach is that errors in training data can be corrected using the information provided by the 3D LiDAR-based tracking. In order to do this, an efficient 3D cluster detector of potential human targets has been implemented. We evaluate the framework using a new 3D LiDAR dataset of people moving in a large indoor public space, which is made available to the research community. The experiments analyse the real-time performance of the cluster detector and show that our online learned human classifier matches and in some cases outperforms its offline version.**

I. INTRODUCTION

In service robotics, detecting and tracking moving objects is key to implementing useful and safe robot behaviors. Identifying which of the detected objects are humans is particularly important for domestic and public environments. Typically the robot is required to collect environmental data of the surrounding area using its on-board sensors, analysing where humans are and where they are going to. Humans should be detected and tracked accurately and as early as possible in order to have enough time to react accordingly. Unfortunately many service robots trade accuracy and robustness of the tracking with the actual coverage area of the detection (i.e. maximum range and field of view of the sensors), which is often limited to a few meters and small angular intervals.

3D LiDAR sensors have recently been applied to many applications in robotics and autonomous vehicles, either alone [1], [2], [3], [4], [5], [6], [7] or in combination with other sensors [8], [9], including human tracking. An important specification of this type of sensor is the ability to provide long-range and wide-angle laser scans. In addition, 3D LiDARs are usually very accurate and not affected by lighting conditions. However, humans are difficult to identify in 3D LiDAR scans because there are no low-level features such as texture and colour, and because of the lack of details when the person is far away from the robot. Detecting features in 3D scans can also be computationally very expensive, as the covered area grows with the range of the sensor, as does the number of human candidates. Moreover, previous methods mostly apply an offline learned classifier for human detection, which usually requires a large number of manually-annotated training data. Unfortunately, labelling this data is tedious work that is prone to human error. Such an approach is also infeasible when dealing with very complex real-world scenarios and when the same system needs to be (re-)trained for different environments.

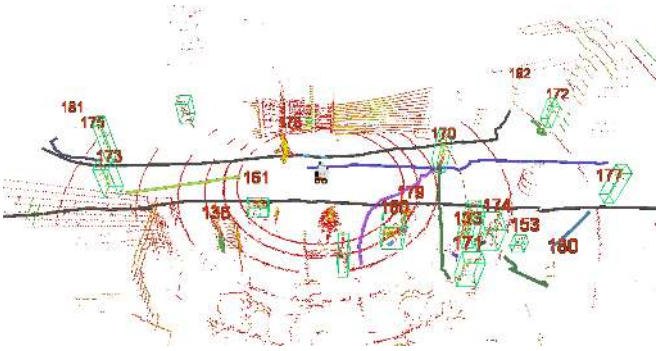


Fig. 1. A screenshot of the 3D LiDAR-based tracking system in action with an online learned human classifier. The detected people are enclosed in green bounding boxes. The colored lines are the people trajectories generated by the tracker.

In this paper, we develop an online learning framework to classify humans from 3D LiDAR detections, taking advantage of and extending our previously developed multi-target tracking system[1] [10] to work with 3D LiDAR scans. We rely on the judgement of a false-positive and a false-negative estimator, similarly to the Positive and Negative "experts" proposed in previous tracking-learning-detection techniques [11], but in this case to train online a classifier that only looks for humans among the detections (i.e. the classification performance does not influence the tracking).

The contributions of this paper are three-fold. First, we present a computationally efficient clustering algorithm for 3D LiDAR scans suitable for real-time model-free detection and tracking. Then, we propose a framework for online learning of a human classifier, which estimates the classifier's errors and updates it to continually improve its performance. Finally, we provide a large dataset[2] of partially-labeled 3D LiDAR point clouds to be used by the research community for training and comparison of human classifiers. This dataset captures new research challenges for indoor service robots

[1] https://github.com/lcas/bayestracking
[2] https://lcas.lincoln.ac.uk/wp/research/data-sets-software/l-cas-3d-point-cloud-people-dataset/

including human groups, children, people with trolley, etc.

The remainder of this paper is organized as follows. Section II gives an overview of the related literature, in particular about 3D LiDAR-based human detection and tracking. Then, we introduce our framework in Section III and the link between tracking and online learning. The former is presented in Section IV, including a detailed description of the 3D cluster detection. The actual online learning is explained in Section V, which clarifies the role of the P-N experts in the classification improved by tracking. Section VI presents the experimental setup and results, as well as our new 3D LiDAR dataset. Finally, conclusions and future research are discussed in Section VII.

II. RELATED WORK

Human detection and tracking have been widely studied in recent years. Many popular approaches are based on RGB-D cameras [12], [13], although these have limited range and field of view. 3D LiDARs can be an alternative, but one of the main challenges working with these sensors is the difficulty of recognizing humans using only the relatively low information they provide. A possible approach to detect humans is by clustering point clouds in depth images or 3D laser scans. For example, Rusu [14] presented a straightforward but computationally expensive method based on Euclidean distance. Bogoslavskyi and Stachniss [15] proposed a faster approach, although the computational efficiency limits the clustering precision. In our method, instead, both runtime and precision are opportunely balanced.

A very common approach is to use an offline trained classifier for human detection. For example, Navarro-Serment *et al.* [1], introduced seven features for human classification and trained an SVM classifier based on these features. Kidono *et al.* [3] proposed two additional features considering the 3D human shape and the clothing material (i.e. using the reflected laser beam intensities), showing significant classification improvements. Li *et al.* [7] implemented instead a resampling algorithm in order to improve the quality of the geometric features proposed by the former authors. Spinello *et al.* [4] combined a top-down classifier based on volumetric features and a bottom-up detector, to reduce false positives for distant persons tracked in 3D LiDAR scans. Wang and Posner [8] applied a sliding window approach to 3D point data for object detection, including humans. They divided the space enclosed by a 3D bounding box into sparse feature grids, then trained a linear SVM classifier based on six features related to the occupancy of the cells, the distribution of points within them, and the reflectance of these points. The problem with offline methods, though, is that the classifier needs to be manually retrained every time for new environments.

The above solutions rely on pre-trained classifiers to detect humans from the most recent LiDAR scan. Only a few methods have been proposed that use tracking to boost human detection. Shackleton *et al.* [2], for example, employed an Extended Kalman Filter to estimate the position of a target and assist human detection in the next LiDAR scan. Teichman *et al.* [5] presented a semi-supervised learning method for track classification. Their method requires a large set of labeled background objects (i.e. no pedestrians) to trains classifiers offline, which showed good performances for track classification but not for object recognition. Our solution, instead, simultaneously learns human and background, and iteratively corrects classification errors online.

Besides datasets collected with RGB-D cameras [12], [13], [16], there are a few 3D LiDAR datasets available to the scientific community for outdoor scenarios [4], [6], [9], [17], but not with annotated data for human tracking in large indoor environments, like the one presented here.

Some authors proposed annotation-free methods. Deuge *et al.* [6] introduced an unsupervised feature learning approach for outdoor object classification by projecting 3D LiDAR scans into 2D depth images. Dewan *et al.* [18] proposed a model-free approach for detecting and tracking dynamic objects, which relies only on motion cues. These methods, however, are either not very accurate or unsuitable for slow and static pedestrians.

It is clear that there remains a large gap between the state of the art and what would be required for an annotation-free, high-reliability human classification implementation that works with 3D LiDAR scans. Our work helps to close this gap by demonstrating that human classification performance can be improved by combining tracking and online learning with a mobile robot in highly dynamic environments.

III. GENERAL FRAMEWORK

Our learning framework is based on four main components: a 3D LiDAR point cluster detector, a multi-target tracker, a human classifier and a sample generator (see Fig. 2). At each iteration, a 3D LiDAR scan (i.e. 3D point cloud) is first segmented into clusters. The position and velocity of these clusters are estimated in real-time by a multi-target tracking system, which outputs the trajectories of all the clusters. At the same time, a classifier identifies the type of cluster, i.e. human or non-human. At first, the classifier has to be initialised by supervised training with labeled clusters of human subjects. The initial training set can be very small though (e.g. one sample), as more samples will be incrementally added and used for retraining the classifier in future iterations.

The classifier can make two types of errors: false positive and false negative. Based on the estimation of the error type by two independent "experts", i.e. a positive P-expert and a negative one N-expert, which cross-check the output of the classifier with that of the tracker, the sample generator produces new training data for the next iteration. In particular, the P-expert converts false negatives into positive samples, while the N-expert converts false positives into negative samples. When there are enough new samples, the classifier is re-trained. The process typically iterates until convergence (i.e. no more false positives and false negatives) or some other stopping criterion is reached.

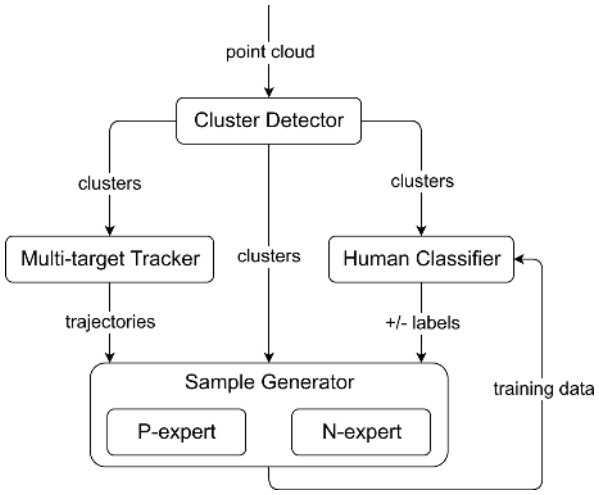


Fig. 2. Process details of the online learning framework.

Our system, however, differs from the previous work [11] in three key aspects, namely the frequency of the training process, the independence of the tracker from the classifier, and the implementation of the experts. In particular, rather than instance-incremental training (i.e. frame-by-frame training), our system relies on less frequent batch-incremental training [19] (i.e. gathering samples in batches to train classifiers), collecting new data online as the robot moves in the environment. Also, while the performance of the human classifier depends on the reliability of the P-N experts and the tracker, the latter is independent from and completely unaffected by the classification performance. Finally, the implementation of our experts can deal with more than one target and therefore generate new training samples from multiple detections, speeding up the online training process.

Note that under particular conditions, the stability of our training process is guaranteed as per [11]. The assumption here is that the number of correct samples generated by the N-expert is greater than the number of errors of the P-expert, and conversely that the correct samples of the P-expert outnumber the errors of the N-expert. Although in the real world these assumptions are not always met, the stability of our system is simplified by the fact that we operate in environments where the vast majority of moving targets are humans and occasional errors are corrected online.

IV. 3D LIDAR-BASED TRACKING

Key components of this system include the efficient 3D LiDAR point cluster detector and the robust multi-target tracker. This section provides details about both.

A. Cluster Detector

The input of this module is a 3D LiDAR scan, which is defined as a set of I points:

$$P = \{p_i \mid p_i = (x_i, y_i, z_i) \in \mathbb{R}^3, i = 1, \ldots, I\} \quad (1)$$

The first step of the cluster detection is to remove the ground plane by keeping only the points p_i with $z_i \geq z_{min}$,

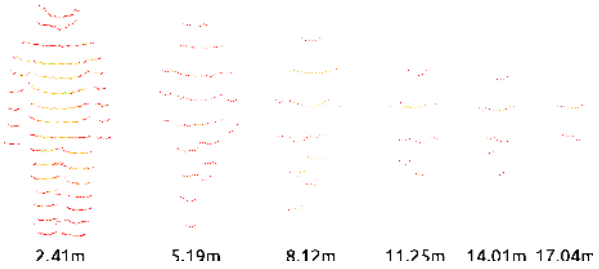


Fig. 3. Examples of a human (1.68 m high) detected by a 3D LiDAR at different distances.

obtaining a subset $P^* \subset P$. This is necessary in order to remove from object clusters points that belong to the floor, accepting the fact that small parts of the object bottom could be removed as well. Note that this simple but efficient solution works well only for relatively flat ground, which is one of the assumptions in our scenarios.

Point clusters are then extracted from the point cloud P^*, based on the Euclidean distance between points in 3D space. A cluster can be defined as follows:

$$C_j \subset P^*, j = 1, \ldots, J \quad (2)$$

where J is the total number of clusters. A condition to avoid overlapping clusters is that they should not contain the same points [14], that is:

$$C_j \cap C_k = \emptyset, \text{ for } j \neq k, \text{ if } min\|p_j - p_k\|_2 \geq d^* \quad (3)$$

where the sets of points $p_j, p_k \in P$ belong to the point clusters C_j and C_k respectively, and d^* is a distance threshold.

Accurate cluster extraction based on Euclidean distance is challenging in practice. If the value of the distance threshold d^* is too small, a single object could be split into multiple clusters. If too high, multiple objects could be merged into one cluster. Moreover, in 3D LiDAR scans, the shape formed by laser beams irradiated on the human body can be very different, depending on the distance of the person from the sensor (see Fig. 3). In particular, the vertical distance between points can vary a lot due to the vertical angular resolution, which is usually limited for this type of sensor. We therefore propose an adaptive method to determine d^* according to different scan ranges, that can be formulated as:

$$d^* = 2\ r\ \tan\frac{\Theta}{2} \quad (4)$$

where r is the scan range of the 3D LiDAR and Θ is the fixed vertical angular resolution. In practice, d^* is the vertical distance between two adjacent laser scans. Obviously, the farther the person from the sensor, the larger is the gap between the vertical laser beams, as depicted in Fig. 3. In the case of our sensor, for example, the resolution is 2° (while horizontally the angular resolution is much smaller). This means that to cluster points at a distance of, for example, 9 m, the minimum threshold should be $d^* = 0.314\ m$.

Clustering points in 3D space, however, can be computationally intensive. The computational load is proportional to the desired coverage area: the longer the maximum range,

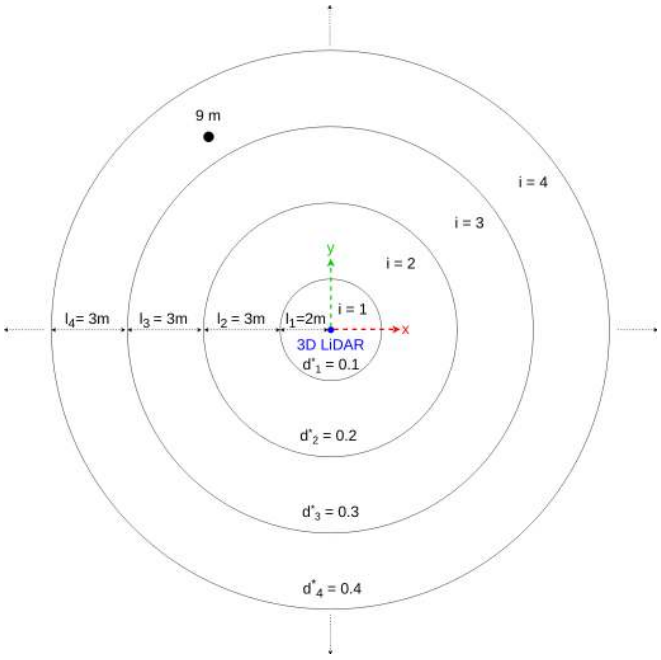


Fig. 4. Different values of d^* correspond to different nested regions.

the higher the value of d^*, and therefore the number of point clouds that can be considered as clusters. In addition, the larger the area, the more likely it is that indeed new clusters will appear within it. To face this challenge, we propose to divide the space into nested circular regions centred at the sensor (see Fig. 4), like wave fronts propagating from a point source, where different distance thresholds are applied. In practice, we consider a set of values d_i^* at fixed intervals Δd, where $d_{i+1}^* = d_i^* + \Delta d$. For each of them, we compute the maximum cluster detection range r_i using the inverse of Equation (4), and round them down to obtain the radius $R_i = \lfloor r_i \rfloor$ of the circular area. The area corresponding to d_i^* is therefore the ring with width $l_i = R_i - R_{i-1}$, where R_0 is just the centre of the sensor. Using $\Delta d = 0.1$ m, we define rings 2-3 m wide, depending on the approximation, which is a good resolution to detect potential human clusters. In the example considered above, a cluster at 9 m from the sensor would belong to the 4th ring, where a threshold $d_4^* = 0.4$ m would be applied.

Finally, a human-like volumetric model is used to filter out over- and under-segmented clusters:

$$\begin{aligned} \overline{C} \;=\; & \{C_j \mid 0.2 \leq w_j \leq 1.0, \\ & 0.2 \leq d_j \leq 1.0,\ 0.2 \leq h_j \leq 2.0\} \end{aligned} \tag{5}$$

where w_j, d_j and h_j represent, respectively, the width, depth and height (in meters) of the volume containing C_j.

B. Multi-target Tracker

Cluster tracking is performed using Unscented Kalman Filter (UKF) and Nearest Neighbour (NN) data association methods, which have already been proved to perform efficiently in previous systems [10], [16]. Tracking is performed in 2D, assuming people move on a plane, and without taking into account the 3D cluster size, which is left to future extensions of our work.

The estimation consists of two steps. In the first step, the following 2D constant velocity model is used to predict the target state at time t_k given the previous state at t_{k-1}:

$$\begin{cases} x_k = x_{k-1} + \Delta t\ \dot{x}_{k-1} \\ \dot{x}_k = \dot{x}_{k-1} \\ y_k = y_{k-1} + \Delta t\ \dot{y}_{k-1} \\ \dot{y}_k = \dot{y}_{k-1} \end{cases} \tag{6}$$

where x and y are the Cartesian coordinates of the target, $\dot{x}$ and $\dot{y}$ the respective velocities, and $\Delta t = t_k - t_{k-1}$. In the second step, if one or more new observations are available from the cluster detector, the predicted states are updated using a 2D polar observation model:

$$\begin{cases} \theta_k &= tan^{-1}(y_k/x_k) \\ \gamma_k &= \sqrt{x_k^2 + y_k^2} \end{cases} \tag{7}$$

where θ_k and γ_k are, respectively, the bearing and the distance of the cluster from the detector, extracted from the projection on the (x, y) plane of the cluster's centroid:

$$c_j = \frac{1}{|C_j|} \sum_{p_i \in C_j} p_i \tag{8}$$

For the sake of simplicity, in the above equations, noises and transformations between robot and world frames of reference are omitted. However, it is worth noting that, from our experience, the choice of the (non-linear) polar observation model, rather than a simpler (linear) Cartesian model, as in [20], is important for the good performance of long range tracking. This applies independently of the robot sensor used, as in virtually all of them, the resolution of the detection decreases with the distance of the target. In particular, the polar coordinates better represent the actual functioning of the LiDAR sensor, so its angular and range noises are more accurately modelled. This leads also to the UKF adoption, since it is known to perform better than Extended Kalman Filters (EKF) in the case of non-linear models [10]. Finally, the NN data association takes care of multiple cluster observations in order to update, in parallel, multiple UKFs (i.e. one for each tracked target).

V. ONLINE LEARNING FOR HUMAN CLASSIFICATION

Online learning is performed iteratively by selecting and accumulating a pre-defined number of new cluster samples while the robot moves and/or tracks people, and re-training a classifier using old and new samples. Details of the process are presented next.

A. Human Classifier

A Support Vector Machine (SVM) [21] is used for human classification, which is known to be effective in non-linear cases and has shown to work well experimentally in 3D LiDAR-based human detection [1], [3]. Six features with a total of 61 dimensions are extracted from the clusters for

TABLE I

FEATURES FOR HUMAN CLASSIFICATION

Feature	Description	Dimension
f_1	Number of points included in the cluster	1
f_2	Minimum cluster's distance from the sensor	1
f_3	3D covariance matrix of the cluster	6
f_4	Normalized moment of inertia tensor	6
f_5	Slice feature for the cluster	20
f_6	Reflection intensity's distribution (mean, standard dev. and normalized 1D histogram)	27

human classification, as shown in Table I. The set of feature values of each sample C_j forms a vector $f_j = (f_1, \ldots, f_6)$. Features from f_1 to f_4 were introduced by [1], while features f_5 and f_6 were proposed by [3]. We discard the other three features, i.e. the so-called "geometric features", presented in [1], because of their relatively low classification performance [3] and the heavy computational load observed in our experiments, which make them unsuitable for real-time tracking. We also observed that our classifier, based on this set features, can typically identify both standing and sitting people, even after being initially trained with samples of walking people only.

A binary classifier is trained for human classification (i.e. human or non-human) at each iteration, based on the above features, using LIBSVM [22]. The ratio of positive to negative training samples is set to $1:1$, and all data are scaled to $[-1, 1]$, generating probability outputs and using a Gaussian Radial Basis Function kernel [23]. Since LIBSVM does not currently support incremental learning, our system stores all the training samples accumulated from the beginning and retrains the entire classifier at each new iteration. The framework, however, also allows for other classifiers and learning algorithms.

B. Sample Generator

An approach based on two independent positive and negative experts is adopted for generating new training samples. At each time step, the P-expert analyses all the new cluster samples classified as negative, identifies those that are more likely to be wrong (i.e. false negatives) and adds them to the training set as positive samples. The N-expert instead analyses samples classified as positive, extracts the wrong ones (i.e. false positives) and adds them to the set of negative samples for the next training iteration. The P-expert increases the classifier's generality, while the N-expert increases the classifier's discriminability. Once a pre-defined number of new samples is collected, the augmented training set is used to re-train the classifier. This learning process iterates until convergence or other stopping criterion, such as maximum training set size.

The P-expert is based on the tracker's trajectories. The idea is that clusters classified as non-human (negative) but belonging to a human-like trajectory in which at least one cluster has been classified as human (positive), will be considered as false negatives and added to the training set as positive samples. In our system, a human-like trajectory satisfies the following two conditions: 1) the target moves a minimum distance r^p_{min} within a given time interval $K\ \Delta t$:

$$r_k = \sqrt{(x_k - x_{k-1})^2 + (y_k - y_{k-1})^2} \quad \text{and} \quad \sum_{k=1}^{K} r_k \geq r^p_{min} \tag{9}$$

and 2) the target's velocity is non-zero but also not faster than a person's preferred walking speed of 1.4 m/s [24]:

$$v_k = \sqrt{\dot{x}_k^2 + \dot{y}_k^2} \quad \text{and} \quad v^p_{min} \leq v_k \leq v^p_{max} \tag{10}$$

In addition, a human-like sample is selected only if the variances (σ_x^2, σ_y^2) of its estimated position (x_k, y_k) satisfy the following condition:

$$\sigma_x^2 + \sigma_y^2 \leq (\sigma^p_{max})^2 \tag{11}$$

The values of K, r^p_{min}, v^p_{min}, v^p_{max}, and σ^p_{max} are empirically determined. The last threshold, in particular, filters out objects (true negatives) that are associated to human-like trajectories but are too "uncertain" because moving in an unexpected way or affected by the proximity of other clusters (see Fig. 5).

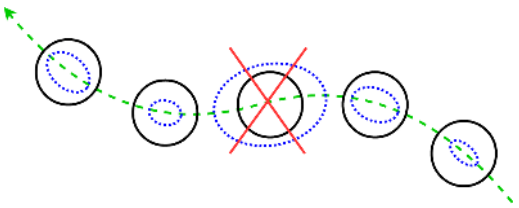


Fig. 5. Example of human-like trajectory samples, including one (red-crossed) filtered out because too uncertain. The green dashed line is the target's trajectory, while the blue dashed circles are the position's uncertainties.

The N-expert converts false positives into new negative samples. We assume that people are not completely static, and there will still be some small changes in the clusters' shape and/or position even though they are just standing or sitting. Taking advantage of the 3D LiDAR's high accuracy, these static objects with low position variances can be identified by the following conditions:

$$r_k \leq r^n_{max} \text{ and } v_k \leq v^n_{max} \text{ and } \sigma_x^2 + \sigma_y^2 \leq (\sigma^n_{max})^2 \tag{12}$$

The parameters r^n_{max}, v^n_{max}, and σ^n_{max} are determined empirically. In practice, the N-expert selects those clusters that were originally classified as humans, although belonging to other static objects (false positives), and adds them to the training set as negative samples.

VI. EXPERIMENTS

A. Dataset

We evaluated the framework on a new dataset collected with a Velodyne VLP-16 3D LiDAR in one of the main buildings of our university. The 3D LiDAR has 16 scan channels with a 360° horizontal and 30° vertical field-of-view, and was mounted at a height of 0.8 m from the floor on the top of a Pioneer 3-AT robot, as shown in Fig. 6. It was

Fig. 6. Robot equipped with (1) a Velodyne VLP-16 3D LiDAR used for dataset collection.

set to rotate at 10 Hz with a maximum scan range of 100 m for data recording. The dataset includes 28,002 scan frames recorded with the robot both while it was stationary and moving in the building. Each frame contains around 30,000 3D points. The robot odometry, coordinate transformation, as well as panoramic image surrounding the robot were also recorded, providing a complete ground-truth for algorithm evaluation. The dataset captures many challenges, such as human groups, children, people with trolleys, etc., as shown in Fig. 7, which are not addressed by most of the current solutions.

A set of 5,492 frames (about 19.6% of the total) was manually annotated using a new open-source GUI tool[3], which contains 6,140 single-person labels ("pedestrian"). The minimum and maximum number of 3D points included in the single-person labels are 3 and 3,925 respectively, while the minimum and the maximum distance from the sensor to the single-person labels are 0.5 m and 27.0 m respectively.

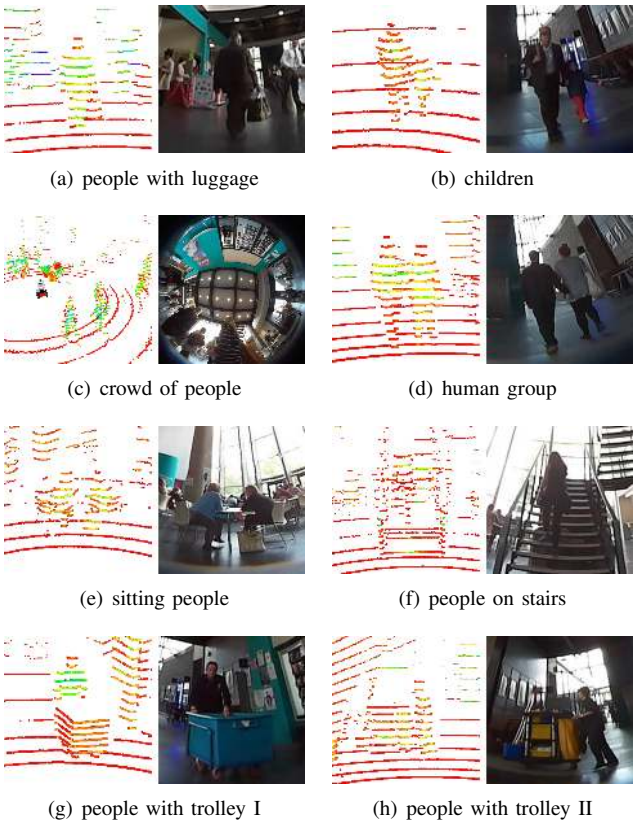
B. Experimental Setup

Our framework has been fully implemented into the Robot Operating System (ROS) [25] with high modularity. All components are ready for download[4] and use by other researchers. Dataset collection, as well as all experiments reported in this paper, were carried out with Ubuntu 14.04 LTS (64-bit) and ROS Indigo, with an Intel i7-4785T processor and 8 GB memory. The data were recorded in sensor frame of references, and the transformation between the coordinate frames was implemented by the ROS *tf* package.

C. Clustering Performance

Previous studies have shown that real-time human tracking can be performed successfully when the sensor update rate is ≥ 5 Hz [10]. In this experiments, we ran the detector over the entire dataset and observed its operating frequency with respect to different detection distances. Fig. 8 shows that the performance of our cluster detector meets the desired update

[3]https://github.com/lcas/cloud_annotation_tool
[4]https://github.com/lcas/online_learning



(a) people with luggage (b) children

(c) crowd of people (d) human group

(e) sitting people (f) people on stairs

(g) people with trolley I (h) people with trolley II

Fig. 7. Different challenges captured in our dataset.

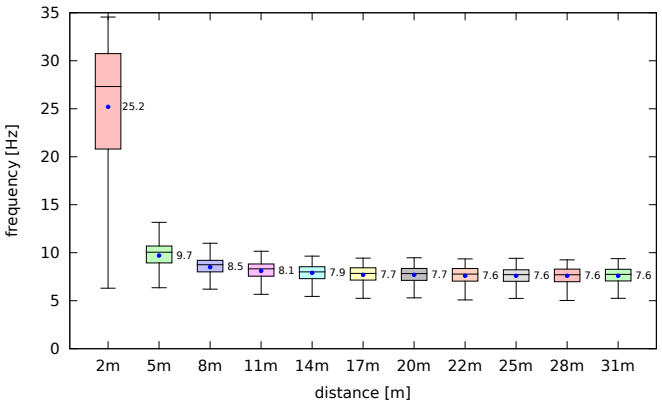


Fig. 8. Clustering performance over a range of distances, with mean shown on the right of each box.

rate requirement. The results show that average frequency decreases with the cluster distance, becoming steady from 22 m onwards. The detection speed, however, was always enough for real-time people tracking, and could be further increased by simply reducing the maximum detection range.

It is worth noting that the maximum number of moving targets simultaneously tracked was 17. Taking advantage of our cluster detector, the maximum distance from the sensor to a tracked moving target was approximately 25 m. Also, thanks to the 360° horizontal field-of-view of the 3D LiDAR, it was possible to track the same target continuously for more than 40 m (total path length), corresponding to a linear

TABLE II
PERFORMANCE ANALYSIS OF THE P-N EXPERTS

Iteration	P	TP	FP	N	TN	FN
1	175	175	0	0	0	0
2	174	174	0	39	39	0
3	156	151	5	5	0	5
4	113	110	3	0	0	0
5	49	49	0	25	0	25
6	183	183	0	130	84	46
7	41	41	0	0	0	0
8	122	121	1	0	0	0
9	123	123	0	0	0	0
10	108	106	2	93	93	0
11	102	101	1	29	29	0
12	129	129	0	0	0	0
13	67	67	0	185	55	130
14	37	34	3	0	0	0
15	63	63	0	0	0	0
16	115	110	5	0	0	0
17	57	57	0	0	0	0
18	33	31	2	30	30	0
19	41	41	0	0	0	0
20	37	30	7	0	0	0
Total	**1952**	**1896**	**56**	**536**	**330**	**206**

P-precision	97.1%	**N-precision**	61.6%

displacement of almost 30 m.

D. Experts Performance

Table II reports the performance of our P-N experts for 20 learning iterations. We count all positives (P), true positives (TP), false positives (FP), all negatives (N), true negatives (TN), and false negatives (FN) in every iteration and report the total numbers and the precision of each expert. The P-expert shows a high precision, since the moving objects in our dataset are mostly humans. The N-expert precision is lower, because our assumption that "people are not completely static" does not always hold. For instance, in iteration 13 the number of errors of the N-expert was relatively large because a person, who was standing close to a static object for a long time, was considered a negative sample. However, the results illustrate that the total number of errors of the P-expert (56) is far less than the number of correct samples generated by the N-expert (330), and the P-expert's correct samples (1896) are far more than the N-expert's errors (206), which satisfy the stability criteria discussed in Sec. III.

E. Classification Results

A comparison of classification performance between an offline trained classifier and the online learned classifier was conducted. In order to illustrate the evolution of the online learned one, both initial and final classifiers were evaluated. The offline classifier was trained using the annotated data, i.e. 6,140 single-person samples, with an equal amount of randomly selected negative samples (non-human). The online initial classifier was trained by a human supervisor with 100 positive samples, plus an equal amount of randomly selected negative ones. The online classifier was then retrained every 300 positive and 300 negative samples, until 6,140 positives and 6,140 negatives had been acquired.

For the test set, we selected 100 scan frames from the dataset distributed across 18 minutes (excluding those frames already manually annotated) and fully annotated these, including standing and sitting people. This contains 995 single-person labels with point cluster size varying from 5 to 2,250, and distance from the sensor between 0.7 m to 19.9 m. The classification performance was evaluated using Precision, Recall, Average Precision (AveP) [26] and F-measure. A true positive was considered such if the overlap between the ground truth and the detection was larger than 50%. Experimental results are shown in Fig. 9. The results illustrate that the final classifier obtained a great improvement by online learning with respect to the initial one. Moreover, the final online classifier matched and in some cases outperformed the offline trained classifier, also thanks to the fact that our online learning framework facilitates the detection of many long-distance samples provided by the tracker, which are difficult to label instead by a human annotator.

VII. CONCLUSIONS

In this paper, we presented an online learning framework for human classification from 3D LiDAR scans. The framework, which relies on a robust multi-target tracking system, enables a mobile robot to learn what humans look like directly from the deployment environment, greatly reducing the need for data annotation. Inspired by previous P-N learning methods, two tracking-based experts have been developed in order to correct errors made by the classifier at each learning iteration. The experimental results based on a real-world dataset demonstrate that the classification performance has been significantly improved.

The proposed framework works in real-time and has been fully implemented in ROS with a high level of modularity. The software and the dataset are publicly available to the research community to perform objective and systematic comparisons between the classification capabilities of different robots. Moreover, our framework should be easy to extend to other moving objects such as cars, bicycles, and animals, or to other sensors such as RGB-D cameras and 2D LiDAR.

Future work will include extending the approach to improve human detection and tracking with the online learned classifier, to better disambiguate different clusters (e.g. shorter versus taller people) by tracking human sizes, and to fuse other sensors to better deal with challenging situations such as groups and strong occlusions.

ACKNOWLEDGMENT

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 645376 (FLOBOT).

REFERENCES

[1] L. E. Navarro-Serment, C. Mertz, and M. Hebert, "Pedestrian detection and tracking using three-dimensional ladar data," in *Proceedings of the 7th Conference on Field and Service Robotics (FSR)*, 2009, pp. 103–112.

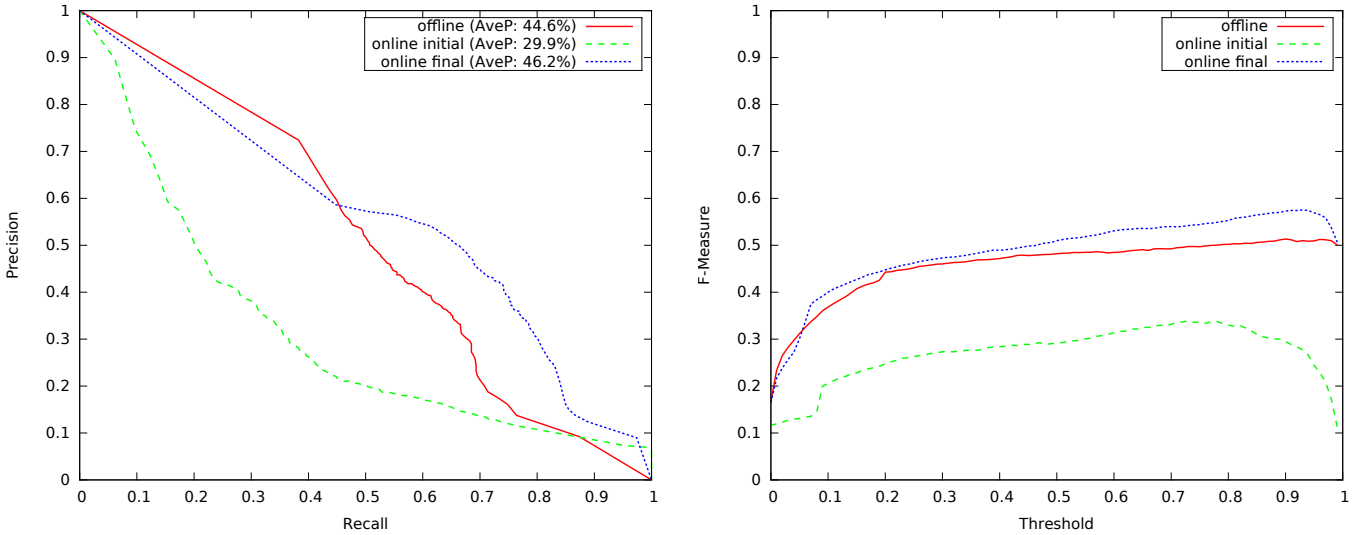


Fig. 9. Performance evaluation of offline and online human classifiers.

[2] J. Shackleton, B. V. Voorst, and J. A. Hesch, "Tracking people with a 360-degree lidar," in *Proceedings of the Seventh IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2010, pp. 420–426.

[3] K. Kidono, T. Miyasaka, A. Watanabe, T. Naito, and J. Miura, "Pedestrian recognition using high-definition LIDAR," in *Proceedings of the 2011 IEEE Intelligent Vehicles Symposium (IV)*, 2011, pp. 405–410.

[4] L. Spinello, M. Luber, and K. O. Arras, "Tracking people in 3d using a bottom-up top-down detector," in *Proceedings of the 2011 IEEE International Conference on Robotics and Automation (ICRA)*, 2011, pp. 1304–1310.

[5] A. Teichman and S. Thrun, "Tracking-based semi-supervised learning," *International Journal of Robotics Research (IJRR)*, vol. 31, no. 7, pp. 804–818, 2012.

[6] M. D. Deuge, A. Quadros, C. Hung, and B. Douillard, "Unsupervised feature learning for classification of outdoor 3d scans," in *Proceedings of the Australasian Conference on Robotics and Automation (ACRA)*, 2013.

[7] K. Li, X. Wang, Y. Xu, and J. Wang, "Density enhancement-based long-range pedestrian detection using 3-d range data," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, pp. 1368–1380, 2016.

[8] D. Z. Wang and I. Posner, "Voting for voting in online point cloud object detection," in *Proceedings of Robotics: Science and Systems*, 2015.

[9] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the KITTI vision benchmark suite," in *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012, pp. 3354–3361.

[10] N. Bellotto and H. Hu, "Computationally efficient solutions for tracking people with a mobile robot: an experimental evaluation of bayesian filters," *Autonomous Robots*, vol. 28, pp. 425–438, 2010.

[11] Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-learning-detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, pp. 1409–1422, 2012.

[12] L. Spinello and K. O. Arras, "People detection in RGB-D data," in *Proceedings of the 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2011, pp. 3838–3843.

[13] M. Munaro and E. Menegatti, "Fast RGB-D people tracking for service robots," *Autonomous Robots*, vol. 37, pp. 227–242, 2014.

[14] R. B. Rusu, "Semantic 3D object maps for everyday manipulation in human living environments," Ph.D. dissertation, Computer Science department, Technische Universitaet Muenchen, Germany, 2009.

[15] I. Bogoslavskyi and C. Stachniss, "Fast range image-based segmentation of sparse 3d laser scans for online operation," in *Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 163–169.

[16] T. Linder, S. Breuers, B. Leibe, and K. O. Arras, "On multi-modal people tracking from mobile platforms in very crowded and dynamic environments," in *Proceedings of the 2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 5512–5519.

[17] A. Teichman, J. Levinson, and S. Thrun, "Towards 3d object recognition via classification of arbitrary object tracks," in *Proceedings of the 2011 IEEE International Conference on Robotics and Automation (ICRA)*, 2011, pp. 4034–4041.

[18] A. Dewan, T. Caselitz, G. D. Tipaldi, and W. Burgard, "Motion-based detection and tracking in 3D LiDAR scans," in *Proceedings of the 2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 4508–4513.

[19] J. Read, A. Bifet, B. Pfahringer, and G. Holmes, "Batch-incremental versus instance-incremental learning in dynamic and evolving data," in *Proceedings of the Eleventh International Symposium on Intelligent Data Analysis (IDA 2012)*, 2012, pp. 313–323.

[20] C. Dondrup, N. Bellotto, F. Jovan, and M. Hanheide, "Real-time multisensor people tracking for human-robot spatial interaction," in *Proceedings of the 2015 IEEE International Conference on Robotics and Automation (ICRA), Workshop on Machine Learning for Social Robotics*, 2015.

[21] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[22] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 1–27, 2011.

[23] S. S. Keerthi and C.-J. Lin, "Asymptotic behaviors of support vector machines with gaussian kernel," *Neural Computation*, vol. 15, no. 7, pp. 1667–1689, 2003.

[24] B. J. Mohler, W. B. Thompson, S. H. Creem-Regehr, H. L. Pick, and W. H. Warren, "Visual flow influences gait transition speed and preferred walking speed," *Experimental brain research*, vol. 181, no. 2, pp. 221–228, 2007.

[25] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: an open-source robot operating system," in *ICRA Workshop on Open Source Software*, 2009.

[26] M. Everingham, L. J. V. Gool, C. K. I. Williams, J. M. Winn, and A. Zisserman, "The pascal visual object classes (VOC) challenge," *International Journal of Computer Vision*, vol. 88, pp. 303–338, 2010.