Kinect-based People Detection and Tracking from Small-Footprint Ground Robots

Armando Pesenti Gritti[1] Oscar Tarabini[1] Jérôme Guzzi[2]
Gianni A. Di Caro[2] Vincenzo Caglioti[1] Luca M. Gambardella[2] Alessandro Giusti[2]

***Abstract*— Small-footprint mobile ground robots, such as the popular *Turtlebot* and *Kobuki* platforms, are by necessity equipped with sensors which lie close to the ground. Reliably detecting and tracking people from this viewpoint is a challenging problem, whose solution is a key requirement for many applications involving sharing of common spaces and close human-robot interaction. We present a robust solution for cluttered indoor environments, using an inexpensive RGB-D sensor such as the Microsoft Kinect or Asus Xtion. Even in challenging scenarios with multiple people in view at once and occluding each other, our system solves the person detection problem significantly better than alternative approaches, reaching a precision, recall and F1-score of 0.85, 0.81 and 0.83, respectively. Evaluation datasets, a real-time ROS-enabled implementation and demonstration videos are provided as supplementary material.**

I. INTRODUCTION

Many emerging robotics applications require that mobile robots and humans share common spaces and interact with each other. One of the main issues to be solved to enable these scenarios is reliable detection and tracking of nearby humans. Only after this issue is solved, robots can manage tasks such as the following.

- Safely and efficiently navigate an environment shared with pedestrians [16], following predictable paths that stay out of the pedestrians' way (a form of indirect interaction).
- Adjust their position, following *social rules* that take into account personal spaces [31], in order to facilitate direct interaction and cooperation [23]; an example is a robot moving at an appropriate distance in front of an human with which an interaction is desired or ongoing [34].
- Carry out application-specific tasks such as: monitoring crowd movements; following a specific person in a cluttered environment to provide assistance or predict its intentions [20]; standing in a line with humans [27].

RGB-D sensors, such as the Microsoft Kinect [2] or Asus Xtion [1], are an ideal and widely-adopted tool for human perception: in this work, we deal with the problem of *detecting* and *tracking* one or multiple people when the sensor is mounted on the robot itself. More specifically, we focus on the issues arising from configurations in which the sensor's viewpoint is close to the ground: this is the case of the widely popular Turtlebot and Kobuki robots [5], which we use as a reference platform; however, this is in general true for any small-footprint ground robot, such as autonomous vacuum cleaners, small platforms for surveillance, monitoring, or item delivery.

RGB-D sensors were designed for perceiving people, and many techniques have been published for solving problems much more demanding than detection and tracking, with impressive results [13], [28]. Unfortunately, existing algorithms are not suitable for the task at hand, because in our case one or more key assumptions, such as the visibility of the upper body of the subject in the frame, are not met (we review related literature in Section II). Detecting and tracking people from a low-lying viewpoint is a fundamentally different and mostly unexplored problem with unique challenges (see Figure 3):

- occlusions (by other subjects, other robots, or even short objects such as tables and chairs) happen frequently and are severe;
- when the subject is close to the robot – i.e. just when accurate perception is of paramount importance – most of the subject falls outside of the sensor's field of view; the only visible feature is part of the person's legs, which exhibit a complex, highly-variable appearance with irregular motion and no obvious markers for detection;
- typical indoor environments are cluttered with distractors (such as table legs and chairs) which are not easily distinguishable from human legs;
- the robot's own motion disallows to adopt straightforward techniques for segmenting moving objects.

Our **main contribution** is an efficient technique (Section III) which robustly solves people detection and tracking in this challenging scenario, under the assumption that the floor is planar – which is reasonable for most indoor environments drivable by a ground robot. More specifically, our system detects each person visible in the sensor's field of view and returns its position relative to the robot and its approximate orientation; moreover, each subject is also tracked through time as long as it stays visible (note that we do not deal with the problem of person re-identification after a track is lost for a long time).

In our approach, leg-like objects protruding from the floor are initially segmented, then classified as either human legs

[1]A. Pesenti Gritti, O. Tarabini and V. Caglioti are with DEIB, Politecnico di Milano, Italy.

[2]J. Guzzi, G. A. Di Caro, L. M. Gambardella and A. Giusti are with IDSIA, USI/SUPSI, Lugano, Switzerland.

This research was partially supported by the Swiss National Science (SNSF) Foundation through the National Centre of Competence in Research (NCCR) Robotics (`www.nccr-robotics.ch`).

or distractors by means of a statistical classifier learned from a large, 26000-instance training dataset acquired in 15 different real-world environments; the resulting probabilistic information is processed in a two-level tracking framework, which associates data in space and time to detect and track people. We illustrate our algorithm in detail using real-world datasets, and showcase results in various complex, realistic scenarios (Section IV), representing environments disjoint from those used during training. Quantitative results show significant performance improvements over alternative approaches, highlighting the contributions of the learned classifier and tracking framework. Extensive demonstration videos and a real-time, open-source, ROS-enabled implementation are provided as supplementary material at `http://bit.ly/perceivingpeople`. A preliminary version of our system will be demonstrated to the public in the HRI 2014 video track [35].

II. RELATED WORK

Due to the importance of detection and tracking for practical applications, many solutions have been proposed in the robotics literature.

2D laser range data: Early works mainly exploited 2D laser range data on an horizontal plane: depending on the sensor height, different horizontal sections of the human body are considered, the most common being the waist [15] or the legs [7], [24], [30]. Range measurements are first spatially clustered on the horizontal plane, then each cluster is classified using either machine learning [7] or model fitting algorithms [21]. Approaches based on 2D laser data are efficient, but operate on perceptions limited to one (or few) planar sections of the scene; then, depending on the sensor height, an human and a different object could yield very similar observations and therefore be impossible to discriminate [26]. This yields a reduced accuracy compared to methods based on more powerful sensors, as we verify in Section IV. Moreover, the cheapest 2D laser scanners cost more than 4 times as much as an RGB-D sensor and inexpensive robots such as the Turtlebot and Kobuki are not equipped with one in their default configuration. On the other hand, it's interesting to note that several laser-based approaches [7], [24], [30] share with our system the challenging requirements that arise from adopting a low-lying sensor position, such as dealing with the problem of pairing legs belonging to the same person [6].

RGB-D data: Despite being cheaper than 2D laser scan technology, RGB-D sensors provide aligned color and depth images of the environment in their field of view, allowing to reconstruct a 3D point cloud of the observed space. Furthermore, one may easily merge 2D RGB image processing tecniques with 3D point cloud analysis, which is a key advantage over a monocular video stream, where 3D information is computationally expensive to infer. Since they use structured infrared lighting, RGB-D sensors are not suitable for outdoor applications; in these cases, 3D point clouds of the environment in front of the robot can be instead acquired by means of stereo cameras or 3D laser scanners [33]. Roboticists use RGB-D sensors for disparate indoor perception tasks, ranging from indoor environment mapping [18] to human body pose and gesture recognition [28].

A fundamental issue to be solved when perceiving people is the *detection* problem, i.e., determining whether a person is in the field of view, and its approximate position. Most algorithms dealing with people perception from RGB-D data focus instead on higher-level tasks, such as pose recovery, action or gesture recognition. In these scenarios, a reasonable assumption is that *at least the subject's upper body is visible*: then, the detection problem is solved through simpler techniques such as: looking for obvious person markers (e.g. the head [37]); filtering point clusters matching the expected approximate dimensions of a person (most importantly, the height [9]); or using statistical classifiers trained on the whole shape of the person [32]. However, in order to ensure that the subject's upper body is visible, the viewpoint can not lie too close to the subject; moreover, in order to avoid occlusions, expecially in crowded or cluttered environments, the sensor should stay in an elevated position, at least at the level of the subjects' chest or eyes. The mentioned algorithms, thus, cannot be directly used on small-footprint ground robots, which must by necessity be short, with sensors lying close to the ground, and which may get very close to the subjects. In this case, the detection problem becomes significantly more complicated, due to the uncommon viewpoint, from which the lower part of the human body is the most prominent feature and severe occlusions frequently occur in cluttered scenarios.

Moreover, several RGB-D based methods not developed explicitly for robotics applications assume that the sensor is still, and segment moving objects using background subtraction [4], which is not a suitable approach if the sensor is mounted on a mobile robot, since the whole scene moves w.r.t. the reference frame of the sensor.

A relevant related work [17] faces the problem of people detection in presence of occlusions, by splitting the 3D point cloud in layers according to the height from the floor, then finding clusters in each layer and classifying each cluster as a human segment or not: different clusters classified to be part of an human are finally connected according to their relative distance in order to reconstruct the visible part of the person. However, only occlusions affecting the lower part of the subject are considered (conversely, in our scenario the legs are often the only visible part, which raises a number of challenging issues detailed in the rest of the paper), and each frame is independently processed: instead, our method adopts tracking at different levels of the pipeline, which yields a significant advantage in overall accuracy, which we quantify in Section IV.

To the best of our knowledge, our approach is the first to demonstrate practical and robust detection and tracking of people from an RGB-D sensor mounted on a mobile, low-lying viewpoint.

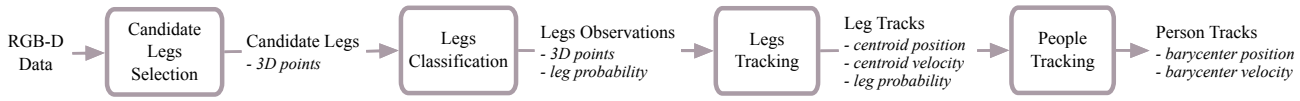


Fig. 1. System overview

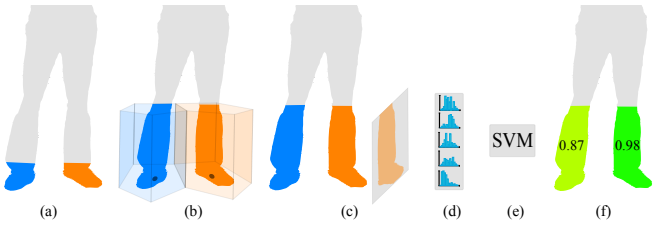


Fig. 2. Point clusters within the first 20 cm above the floor are identified as candidate feet (a). The centroids of the clusters are projected onto the floor plane and are used as generators of a Voronoi diagram. Each candidate foot is then expanded to a candidate leg, by constraining its expansions up to the knee level and only to the points whose vertical projections fall within the corresponding Voronoi region (b). The Rectified Depth Image is computed on each candidate leg (c) and from it the Histogram of Oriented Gradients feature vector is extracted (d). The latter is classified (e) and the corresponding leg probability is obtained (f).

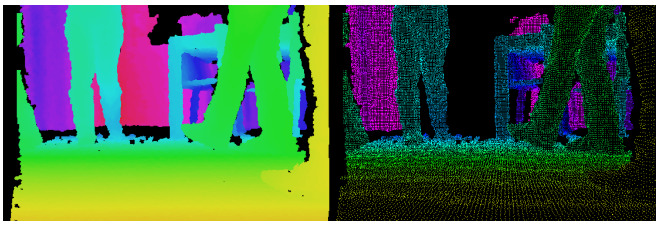


Fig. 3. Original (left) and downsampled (right) point cloud.

III. METHOD

Legs are the most prominent (and, for close subjects, the only) feature which is visible from our considered point of view. Therefore, our approach is based on the detection of legs in the 3D point cloud. However, legs have a very variable appearance, and many objects with a similar shape are present in cluttered environments: therefore, our approach first *detects candidate legs* (Section III-A) using simple rules that yield high sensitivity (i.e. rarely misses an actual leg) but suffer from low specificity (i.e., detects many spurious objects); then, a statistical classifier previously learned from large training datasets is applied to candidates in order to discriminate actual legs from other objects (Section III-B). Individual candidates are tracked along time (Section III-C) and their leg-likelihood continuously updated by integrating classifier outputs in time: at this stage, people can finally be detected and tracked using leg positions and likelihoods as observations in a Kalman filter. An overall view of the system is presented in Figure 1.

A. Data Preprocessing and Selection of Candidate Legs

Initially, point cloud data is preprocessed by discarding points outside the declared range of the sensor (which are affected by large amounts of noise [22]); then, the point cloud is downsampled using a voxel grid approach [3] with a voxel-size of 1 cm. This reduces by more than 80% the amount of data to be processed and yields a more homogeneous point density at different distances from the sensor (Figure 3).

Floor Plane Detection: Because the sensor position and rotation may be only approximately known, we initially detect the floor plane in the 3D point cloud using RANSAC-based [14] plane detection, constraining the search to quasi-horizontal planes. The whole point cloud can now be rigidly transformed in such a way that the floor lies onto the plane $z = 0$, the z-axis points upwards, the y-axis is directed along the projection of the optical axis onto the floor plane and the x-axis is determined to complete a right-handed reference frame.

Candidate Selection: Candidates are identified as point clusters emerging from the floor (Figure 2(a)): in particular, we initially limit the analysis to 3D points within the first 20 cm above the floor and apply a density-based 3D clustering technique. This family of clustering algorithms defines clusters as areas of higher density than the remainder of the data set. The point cloud section we select to apply this search is generally characterized by few sparse objects in large areas: these clear variations in point density perfectly meet the requirements of DBSCAN [12]. The method requires two parameters: a radius delimiting the neighborhood extent and the minimum number of neighbors required to form a cluster. Density-based clustering methods greatly benefit from a relatively homogeneous density in the input point cloud: in our case, this is ensured by the voxel-grid downsampling preprocessing step mentioned above. Indeed this simplifies the setup of the parameters, which we empirically tuned to be a minimum of 5 neighbors in a 4 cm radius, having verified they perform well in very different situations. In practice, this approach may occasionally lead to cluster together two feet, expecially when the person is standing still (conversely, while a person is walking its feet are usually well-spaced and would not be merged): we accept this possibility, and let the subsequent machine learning and tracking approaches handle this case.

The dimensions of the bounding box of each cluster's horizontal projection are used as a simple criterion to pre-filter candidates which are obviously too large or too small to represent either an actual single foot or a joint couple of feet.

Expansion of Candidates to Candidate Legs: The resulting 3D point clusters are limited to a maximum height of 20 cm, and therefore contain little geometric information for discriminating actual feet from other objects; attempting to exploit such data alone would incur in the same issues observed with 2D laser-scans, as many objects would appear

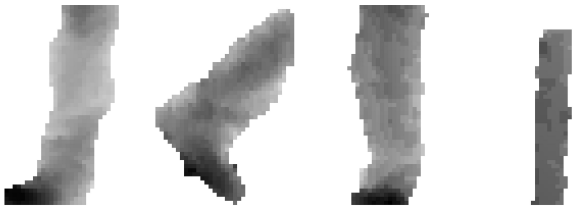


Fig. 4. Grayscale representation of the rectified depth image for four leg candidates, the last of which is a non-leg object.

Fig. 5. Visualization of the results of candidate classification: green and red correspond respectively to high and low leg probability.

similar to feet. Therefore, we now expand candidates up to knee-height (50 cm), thus incorporating useful information about the characterizing features of a person's leg, like the ankle shape and the almost cylindrical surface of the limb. Further expansion would lead to issues with subjects close to the sensor, where anything above the knee would fall outside of the sensor's field of view.

In order to avoid adding too many unrelated points to a candidate leg, which would inevitably occur in case of very close clusters and may lead to an erroneous classification, the expansion of each candidate foot is spatially bounded. In particular, we consider the projection on the floor plane of the centroids of all candidates and compute the 2D Voronoi region for each. For a given candidate, the expansion only considers 3D points whose projection on the floor falls within the respective Voronoi region (Figure 2(b)).

B. Classification

For each candidate 3D shape, a *rectified depth image* is computed and used to build a feature vector for classification by means of a statistical SVM classifier [19].

Rectified Depth Image Computation: The rectified depth image for a given candidate is computed by an orthographic projection of its 3D points onto a vertical plane (Figure 2(c) and 4). Note that, unlike the depth image directly acquired by the sensor, the rectified depth image is not affected by the sensor's pitch and roll (which may not be exactly horizontal), nor by perspective distortions; therefore, it is expected to yield a standardized representation of the candidate's appearance, robust to variations in the sensor's position on the robot. Depth values are computed on a 40x50 cm grid with cell-size of 1 cm, and regularized through mild morphological filtering.

Feature Computation and Classification: We summarize the rectified depth image for a candidate by means of the Histogram of Oriented Gradients [11] descriptor (HOG): this allows us to robustly characterize local 3D shape and appearance of an object by representing local depth changes in simple histograms. A similar descriptor is the Histogram of Depths, which was also adopted for people detection in RGB-D data [32].

We divide the rectified depth image in 4×5 square windows with an edge of 10 cm, then discretize the depth gradients within each window over a 9-bin histogram. This yields a good trade-off between classification accuracy and feature dimensions. The resulting 180-dimensional feature vector is processed by a pre-trained soft SVM binary classifier (with RBF kernel) returning the probability of the *leg* and *non-leg* classes (Figure 2(d,e,f)).

After the classification phase, a set of leg observations, consisting of the leg candidates 3D points together with their leg probability, is available for further processing (Figure 5).

C. Tracking

People are detected and tracked from leg observations computed for each frame by means of a two-step approach: in the first step, individual leg observations are used to robustly detect and track legs in time, obtaining a set of *leg tracks*; in the second step, leg tracks are used as observations in a filter which yields a set of *person tracks*, solving challenging data association issues.

Legs Tracking: Tracking leg observations across frames is not trivial, because temporary occlusions of the legs (by foreground objects, the other leg, or other people) are very frequent. The *state variables* tracked for each leg observation consist of the 2D position and velocity of the leg centroid projected onto the floor plane, as well as its probability of being a leg. The *update step* is performed by associating each leg observation to the closest leg track and using the cluster centroid and the leg probability returned by the classifier as measurements. A constant velocity model is assumed for each leg track, considering random gaussian noise to account for a variable acceleration (in practice, legs have a very irregular movement pattern).

Because the robot may be moving or rotating while sensing, tracking must be performed using a fixed reference frame, not the robot's own: in particular, leg observations are converted from the sensor's reference frame to a fixed reference frame by exploiting the robot's odometry information. We demonstrate this feature in the supplementary video, by showing how tracking is stable to robot rotations, and how a person's velocity is correctly estimated regardless of the robot's own motion.

Tracking multiple objects in cluttered environments is known to be a very challenging task. Several sophisticated algorithms (most notably JPDAF [8], GMPHD [36], MHT [10], and multi-target particle filters [29]) have been proposed to deal with these complex scenarios. We have observed satisfactory results with a simpler and more efficient approach, which accounts for some particular features of the

problem at hand.

In order to associate leg observations to leg tracks, we perform a variant of nearest neighbor data association. In particular, we project on floor the foot points of the leg observation and compute their minimum euclidean distance to the leg track centroid. When this minimum distance is under a small threshold (5 cm), the leg observation is directly associated to that leg track and a classic Kalman Filter update step is performed. If a leg observation can be directly associated to two distinct leg tracks, revealing a consistent possibility that two close legs previously tracked separately have been merged by the clustering phase into a unique leg observation, then both leg tracks are updated, each with a measure corresponding to an estimate of its new centroid. Leg tracks not directly associated to a leg observation are updated through a PDAF [8] approach, accounting for all observations within 15 cm of the track's centroid. The PDAF algorithm extends the standard Kalman Filter in order to perform the track state update in presence of multiple measures, of which the correct association is unknown, also accounting for the fact that none of them may be the target measure. Finally, in the case a leg observation is far (more than 15 cm) from any existing track, a new track is created.

Note that only leg tracks with a leg probability larger than a threshold τ are used for the subsequent level of tracking. τ represents an important parameter of the system, whose impact is evaluated in Section IV.

People Tracking: The second tracking step uses leg tracks as observations and returns a set of person tracks, which is the final output of the system. In particular: the *state variables* tracked for each person consist on the 2D position of its barycenter and its velocity; the *motion model* is assumed to be constant velocity with random gaussian noise; the *measurements* used for updating the state are derived from leg tracks as described below.

In crowded scenarios, correctly associating each leg track to the correct person track is not straightforward: we adopt the following geometric approach.

- First, we find pairs of leg tracks which most likely belong to the same person. In particular, the 3D point cluster associated to each leg track is expanded upwards, to the hip level: if a pair of clusters join, we check whether their barycenter can be directly associated to an existing person track using a nearest-neighbor policy. If this is verified, a classic Kalman state update is applied on the person track, and the two leg tracks are removed from further processing. Note that this procedure misses many leg pairs, expecially for subjects close to the sensor (whose hip area falls outside the sensor's field of view).
- Remaining leg tracks are used to create a list of all leg track pairs which may belong to the same person – i.e. whose mutual distance is shorter than 80 cm. Note that the same leg track may appear in more than one of such pairs. The centroid of each pair is computed and added to a list of *potential person barycenters*. Additionally, the centroid of each individual leg is also included in the same list, in order to account for the frequent case in which a single leg is visible for a person. For each person track yet to be updated, a PDAF approach is performed by accounting for all *potential person barycenters* within a 40 cm radius from the barycenter of the person track.

A critical aspect turns out to be the creation of a new person track. Our system adopts a conservative strategy, creating a new person only when two leg tracks have high classification probabilities, the upward expansions of the associated clusters nearly merge, and their barycenter is far enough from the barycenter of existing person tracks. Consequently, single leg tracks far from any known person track are assumed to belong to a yet untracked person whose second leg has not been observed yet.

IV. EXPERIMENTAL RESULTS

A. Implementation

The prototype system is implemented in MATLAB, with the most computationally expensive tasks written as mex functions able to exploit multi-core CPUs thanks to OpenMP support. Various ROS-Matlab bridges exist [25] which allow to use our prototype within ROS: in particular, it appears as a node listening for RGB-D data and robot odometry, and publishing the positions and velocities of tracked people. Alternatively, the prototype can use the OpenNI library for accessing to live or recorded sensor data. The release is available on-line at `bit.ly/perceivingpeople`. Depending on the scenario complexity, the system processes 10 to 30 fps on a low-end Intel Core i5 laptop. When using a top-end laptop based on an Intel Core i7-4930MX processor, the minimum framerate observed in very complex scenarios raises to 25 fps, whereas for most of the time the full frame rate is achieved with less than 40% CPU load. As we measure in Section IV-F, the system's performance does not significantly degrade even with as few as 10 fps.

B. Dataset Acquisition and Classifier Training

The *training dataset* for the SVM classifier is acquired as follows. We recorded two set of videos: the first set was shot in 15 different cluttered indoor environments without any person appearing in the frame; the second set was shot by placing the sensor in several wide indoor halls, recording only passing people. In the first set of videos, all detected leg candidates have been labeled with ground truth class *non-leg*; in the second set, all leg candidates have ground truth class *leg*. The resulting training dataset contains 26000 candidates, of which 35% are legs.

Moreover, *testing datasets* have been created for evaluating the overall performance of the system. We recorded three 30 seconds RGB-D videos in different and completely new environments (i.e., not included in the training dataset) and manually labelled each single frame, by marking every visible leg and indicating which person it belonged to. The testing scenarios are the following:

S-Easy: The sensor is still and two people randomly walk in an obstacle-free environment.

S-Medium: The sensor is still and three people walk around a small table with few other obstacles in the scene.

S-Difficult: The sensor is on a mobile platform in a quite cluttered environment and a total of three people walk around a small table.

The three testing datasets with associated ground truth are also available as supplementary material, to promote quantitative comparisons with future systems.

C. Performance measures

We tested the performance of our system using the frame-based metrics of *precision (p)*, *recall (r)* and *F1 score (f1)* defined as:

$$p = \frac{TP}{TP + FP} \quad r = \frac{TP}{TP + FN} \quad f1 = \frac{2pr}{p + r} \qquad (1)$$

where: TP is the number of True Positives, FP is the number of False Positives and FN is the number of False Negatives.

We separately evaluated leg detection and person detection.

Legs Detection: This evaluation is intended to measure the performance in correctly identifying legs in each frame, regardless of the person which they belong to. In this case, a True Positive is a correctly detected leg, a False Positive is a leg detection returned by our system which can not be associated to any leg in the ground truth of the frame, and a False Negative is a leg in the ground truth of the frame for which no detection has been reported.

People Detection: This evaluation is intended to express an overall performance measure of our system, which only accounts for whether people visible in the frame are in fact detected. True Positives are counted for each person in the ground truth for which a person track has been returned within a 25 cm radius. False Negatives are counted for each person in the ground truth for which no person track is found within the same limit. False Positives are counted for each detected person track whose associated legs are not in the ground truth, or do not belong to the same person.

D. Alternative methods

In order to evaluate how our system compares with alternative methods, we implemented and tested on the same datasets three different approaches to leg detection.

Inscribed Angles Variance (IAV [21]): a 2D laser-based method that classifies a cluster to be a leg based on geometrical relations.

Supervised Learning on 2D Range Data [7]: a 2D laser-based method which trains a machine learning classifier on 13 cluster features. We compare results obtained with both Adaboost and SVM classifiers.

Histogram of Local Surface Normals (HLSN [17]): an alternative classification feature extracted from 3D points.

For 2D laser-based methods, we synthesize input data by computing an horizontal section of our 3D leg candidates at 30 cm above the floor plane.

E. Quantitative results

We report results computed on the *S-Difficult* scenario in our testing dataset.

Initially, we analyze how the performance changes when tuning the *leg probability threshold* τ, i.e. the probability used to discriminate legs and non-legs candidates. The plot in Figure 6(a) shows the precision-recall curves concerning *Legs Detection* both for our and alternative methods. The *HOG SVM* curve is the output of the classifier used by our system, which clearly outperforms all alternative approaches. Moreover, the *HOG SVM + Tracking* curve shows that tracking also improves frame-based detection performance, because the leg probability associated to each candidate is filtered over time, which limits the impact of occasional occlusions. Moreover, due to the peculiar way data association is handled (see Section III-C), our tracking algorithm manages to return two distinct legs close to each other, even when the preceding clustering algorithm merges both into a single cluster.

The plot in Figure 6(b) shows the precision-recall curves concerning *People Detection*, using both our system, and alternative methods for leg detection. In particular, the performance of a given alternative method is computed by applying our own tracking approach to detections returned by such method.

In order to determine the optimal *leg probability threshold* τ, in Figure 7(a) we analyze the *People Detection* performance of the system while varying τ. We observe that the F1 score reaches a stable maximum level in the interval 0.5 - 0.9, which shows that the parameter is not a critical one. We select an intermediate value of $\tau = 0.8$ for the following experiment. In the same plot we also report the F1 score obtained when tuning τ for the two alternative approaches returning probabilities. In the case of the *Laser SVM* curve, we note that the maximum F1 score is both lower, and less robust to variations in the threshold parameter.

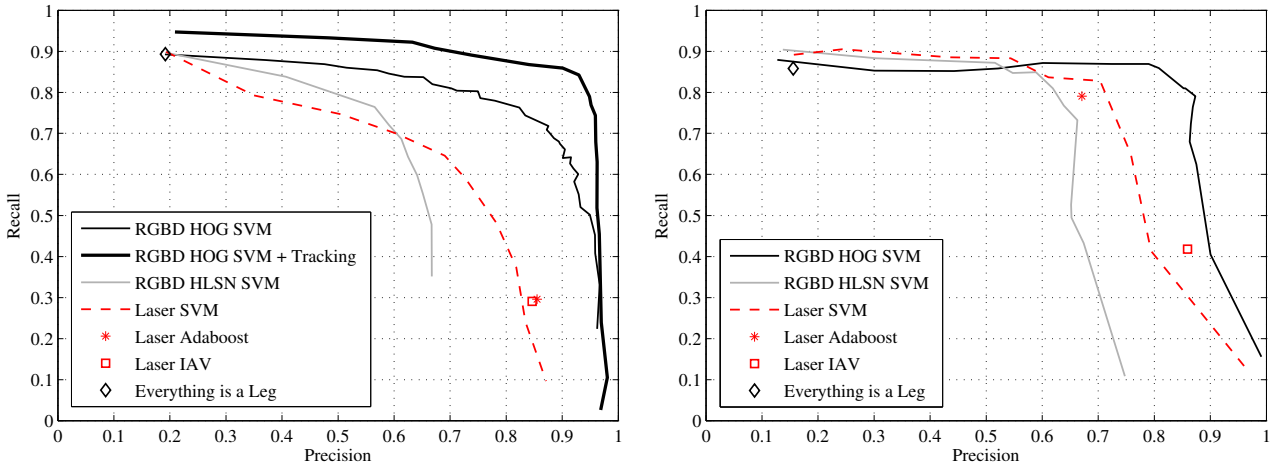
The following table reports the performance of our system on the three different testing scenarios, when using $\tau = 0.8$ as determined above.

	Legs Detection			**People Detection**		
Scenario	*p*	*r*	*f1*	*p*	*r*	*f1*
S-Easy	0.97	0.91	0.94	0.88	0.91	0.89
S-Medium	0.98	0.81	0.88	0.93	0.83	0.88
S-Difficult	0.96	0.79	0.87	0.85	0.81	0.83

Additional qualitative results and considerations are reported in Figure 8.

F. Effect of frame rate for real-time application

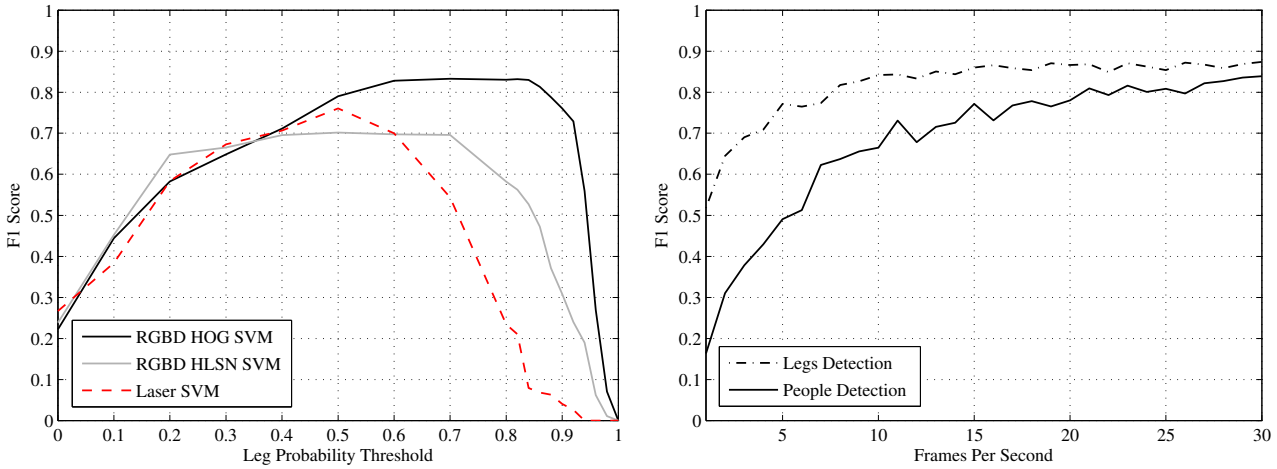
Our tracking approach assumes a constant velocity state transition model. This simplifying hypothesis is a good approximation as long as the temporal interval between two observations remains small. Therefore, we study how the performance changes when limiting the input framerate – which may be useful in case of systems with low processing



(a) Legs Detection Performance Comparison

(b) People Detection Performance Comparison

Fig. 6. Performance in the Precision-Recall plane for different detection approaches. The ideal algorithm has precision = recall = 1 (which yields an F1-score of 1). By tuning the τ parameter, the tradeoff between precision and recall is explored, yielding a precision-recall curve. Some methods appear as single points since by design they only output a binary classification.



(a) People Detection F1 Score Comparison vs τ

(b) Effect of Frame Rate on F1 Score

Fig. 7. F1 score curves. On the left we show how the F1 score changes while tuning τ, for both our system and alternative methods. On the right we plot, for our system, the effect of frame rate on performance using $\tau = 0.8$.

power. Adopting the optimal leg probability threshold determined above, we evaluate our system by simulating a given frame rate on the recorded *S-Difficult* scenario test video. Figure 7(b) reports the F1 score for both the leg detection problem and the people detection problem, versus the simulated frame rate. It can be observed that the performance penalty when limiting the framerate is marginal as soon as the framerate is kept above 10 fps. Under this threshold, the legs tracking performance starts degrading significantly, which also impacts the people tracking performance.

V. DISCUSSION AND CONCLUSIONS

We presented a practical approach for detecting and tracking people indoors, from a small-footprint ground robot using an RGB-D sensor. Solving the problem required careful handling of the peculiar challenges arising from this scenario. We demonstrated that, in all considered datasets, our system performs better than alternatives, and yields good results both quantitatively (Section IV-E) and qualitatively (see supplementary videos at `http://bit.ly/perceivingpeople`). We provided a practical, real-time, ROS-enabled implementation ready to run on common robot platforms such as the Turtlebot and Kobuki. In its current version, the system does not exploit the RGB data provided by the sensor; this is precious information that we plan to exploit for improving data association, and enabling additional functionality such as person re-identification once a track is temporarily lost.

REFERENCES

[1] Asus xtion pro live. `http://bit.ly/19Sfn62`.
[2] Microsoft kinect. `http://bit.ly/1cAhjxI`.
[3] The point cloud library. `http://pointclouds.org/`.
[4] Primesense nite middleware. `http://bit.ly/IAIHRb`.
[5] The turtlebot robot development kit. `http://bit.ly/lqRgND`.

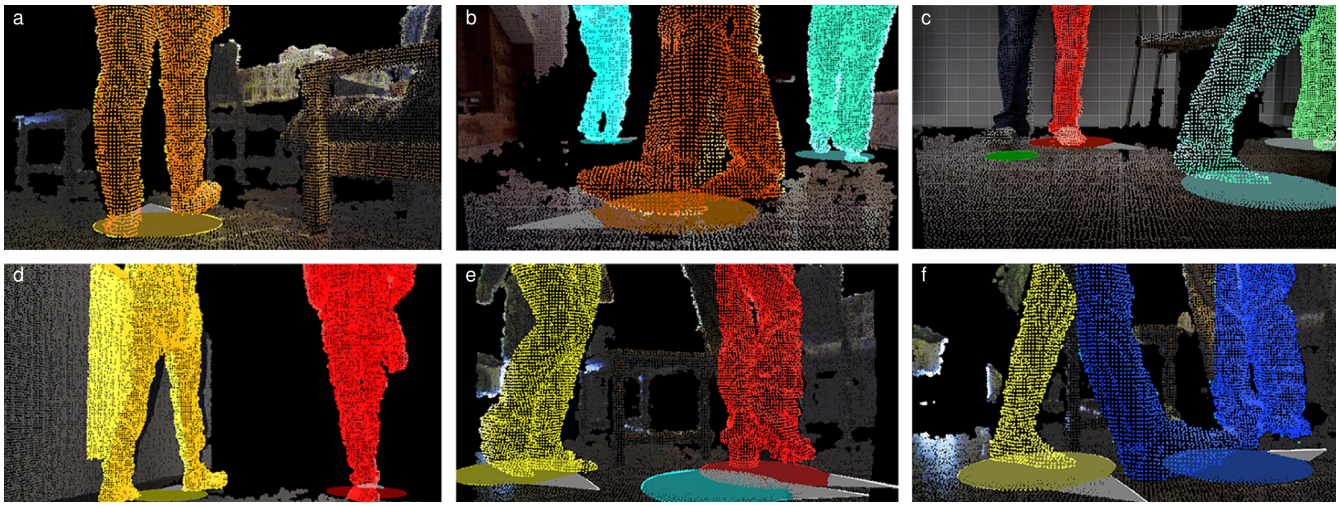


Fig. 8. Example outputs from our system in challenging scenarios. All 3D points corresponding to each detected person are displayed in a different color. A large disk drawn on the floor represents the barycenter for each person, with an arrow pointing towards the computed motion direction. Remaining images show various failure modes. In c) one leg of the person on the left is not joined to the other. In d) part of the wall was merged with the person's leg (which was nonetheless correctly detected). In e) the person on the right was not re-associated to the correct person track (gray disk) after being briefly lost, but a new track was created instead. In f) two legs were not correctly joined, and each was associated to its own person track. Note that most of these failure cases are unstable, short-lived configurations which quickly resolve, as is apparent in supplementary videos.

[6] K. Arras, S. Grzonka, M. Luber, and W. Burgard. Efficient people tracking in laser range data using a multi-hypothesis leg-tracker with adaptive occlusion probabilities. In *Proc. ICRA 2008*.

[7] K. O. Arras, O. M. Mozos, and W. Burgard. Using boosted features for the detection of people in 2d range data. In *Proc. ICRA 2007*.

[8] Y. Bar-Shalom et al. The probabilistic data association filter. *IEEE Control Systems*, 29(6):82–100, 2009.

[9] F. Basso, M. Munaro, S. Michieletto, E. Pagello, and E. Menegatti. Fast and robust multi-people tracking from rgb-d data for a mobile robot. In *Proc. Intelligent Autonomous Systems (IAS) 2012*.

[10] S. Blackman. Multiple hypothesis tracking for multiple target tracking. *Aerospace and Electronic Systems Magazine, IEEE*, 19(1):5–18, Jan 2004.

[11] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proc. CVPR*, 2005.

[12] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proc. Knowledge Discovery and Data Mining (KDD) 1996*.

[13] G. Fanelli, M. Dantone, J. Gall, A. Fossati, and L. Van Gool. Random forests for real time 3d face analysis. *Int. J. Comput. Vision*, 101(3):437–458, February 2013.

[14] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[15] A. Fod, A. Howard, and M. Mataric. A laser-based people tracker. In *Proc. ICRA 2002*.

[16] J. Guzzi, A. Giusti, L. Gambardella, G. Theraulaz, and G. Di Caro. Human-friendly robot navigation in dynamic environments. In *Proc. ICRA 2013*.

[17] F. Hegger, N. Hochgeschwender, G. K. Kraetzschmar, and P. G. Ploeger. People detection in 3d point clouds using local surface normals. In *RoboCup 2012: Robot Soccer World Cup XVI*.

[18] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox. Rgb-d mapping: Using depth cameras for dense 3d modeling of indoor environments. In *Proc. International Symposium on Experimental Robotics (ISER) 2010*.

[19] T. Joachims. Making large scale svm learning practical. 1999.

[20] T. Kanda, D. Glas, M. Shiomi, and N. Hagita. Abstracting people's trajectories for social robots to proactively approach customers. *IEEE Transactions on Robotics*, 25(6):1382–1396, 2009.

[21] H. Kheyruri and D. Frey. Comparison of people detection techniques from 2d laser range data.

[22] K. Khoshelham and S. O. Elberink. Accuracy and resolution of kinect depth data for indoor mapping applications. *Sensors*, 12(2):1437–1454, 2012.

[23] C.-P. Lam, C.-T. Chou, K.-H. Chiang, and L.-C. Fu. Human-centered robot navigation – towards a harmoniously human-robot coexisting environment. *IEEE Transactions on Robotics*, 27(1):99–112, 2011.

[24] J. H. Lee, T. Tsubouchi, K. Yamamoto, and S. Egawa. People tracking using a robot in motion with laser range finder. In *Proc. IROS 2006*.

[25] ROS Wiki Contributors. Ros on matlab. `http://wiki.ros.org/groovy/Planning/Matlab`, 2014.

[26] O. Mozos, R. Kurazume, and T. Hasegawa. Multi-part people detection using 2d range data. *International Journal of Social Robotics*, 2(1):31–40, 2010.

[27] Y. Nakauchi and R. Simmons. A social robot that stands in line. *Autonomous Robots*, 12(3):313–324, 2002.

[28] I. Oikonomidis, N. Kyriazis, and A. A. Argyros. Efficient model-based 3d tracking of hand articulations using kinect. In *Proc. British Machine Vision Conference (BMVC) 2011*.

[29] K. Okuma, A. Taleghani, N. De Freitas, J. J. Little, and D. G. Lowe. A boosted particle filter: Multitarget detection and tracking. In *Proc. European Conference on Computer Vision (ECCV) 2004*.

[30] D. Schulz, W. Burgard, D. Fox, and A. B. Cremers. People tracking with mobile robots using sample-based joint probabilistic data association filters. *The International Journal of Robotics Research*, 22(2):99–116, 2003.

[31] E. Sisbot, R. Alami, T. Simeon, K. Dautenhahn, M. Walters, and S. Woods. Navigation in the presence of humans. In *Proc. IEEE-RAS International Conference on Humanoid Robots (Humanoids) 2005*.

[32] L. Spinello and K. O. Arras. People detection in rgb-d data. In *Proc. IROS 2011*.

[33] L. Spinello, K. O. Arras, R. Triebel, and R. Siegwart. A layered approach to people detection in 3d range data. In *Proc. AAAI 2010*.

[34] M. Svenstrup, S. Tranberg, H. J. Andersen, and T. Bak. Adaptive human-aware robot navigation in close proximity to humans. *International Journal of Advanced Robotic Systems*, 8(2):1–15, 2011.

[35] O. Tarabini et al. Video: Perceiving people from a low-lying viewpoint. In *Proc. Human Robot Interaction (HRI) 2014, to appear*.

[36] B.-N. Vo and W.-K. Ma. The gaussian mixture probability hypothesis density filter. *Signal Processing, IEEE Transactions on*, 54(11):4091–4104, 2006.

[37] L. Xia, C.-C. Chen, and J. Aggarwal. Human detection using depth information by kinect. In *In Proc. Computer Vision and Pattern Recognition Workshops (CVPRW) 2011*.