Autonomous Robots (2023) 47:1255–1273
https://doi.org/10.1007/s10514-023-10125-5

Humans as path-finders for mobile robots using teach-by-showing navigation

Alessandro Antonucci[1] · Paolo Bevilacqua[1] · Stefano Leonardi[1] · Luigi Paolopoli[1] · Daniele Fontanelli[2]

Received: 26 December 2022 / Accepted: 2 July 2023 / Published online: 17 August 2023

Abstract

One of the most important barriers towards a widespread use of mobile robots in unstructured, human populated and possibly a-priori unknown work environments is the ability to plan a safe path. In this paper, we propose to delegate this activity to a human operator that walks in front of the robot marking with her/his footsteps the path to be followed. The implementation of this approach requires a high degree of robustness in locating the specific person to be followed (the *path-finder*). We propose a three phases approach to fulfil this goal: 1. Identification and tracking of the person in the image space, 2. Sensor fusion between camera data and laser sensors, 3. Point interpolation with continuous curvature paths. The approach is described in the paper and extensively validated with experimental results.

Keywords Person-following robot · Human-robot interaction · Human detection and tracking · Service robotics

1 Introduction

When an autonomous mobile robot of remarkable size and mass navigates the treacherous waters of unstructured and human-populated environments, safety concerns and regulation constraints take centre stage and become a barrier for the adoption of this technology. Safety appears as a major challenge in a large class of robotic applications. In industrial settings, mobile robots are required to move within designated areas (Markis et al. 2019). Similarly, in personal and service robotics applications (ISO13482 2014) (e.g., hospitals, museums, hospitals, art galleries), preventing accidents to the humans and to precious assets alike (expensive equipments, precious art-works, etc.) is a precondition for any certified use of robots. To address this challenging problem, we advocate a mixed approach. When the mobile robot travels across a known safe or segregated area, it can move in full autonomy, whilst whenever it enters a shared or dangerous area where, in case, environmental reliable information lacks (e.g., absence of an a-priori map or in a highly dynamic environment), the responsibility of the most critical decisions (i.e., motion planning) is shifted to a human operator.

Our reference scenario can be described as follows. The mobile robot starts its mission with a person standing in front. The robot looks at the person with its visual devices, extracts a number of important features and elects her/him as a *path-finder*. Then starts the second phase: the person walks to the destination, with the robot tracking and following her/him moving along the path marked by her/his footsteps (see Fig. 1a). After the path-finder reaches the destination, the path is memorised and can be used for future missions. Observe that *this is not* a standard leader-follower application in which the robot is allowed to sway sideways as far as it keeps a specified distance from its leader (Lam et al. 2010). In our case, the human is a path-finder and the robot follows exactly her/his virtual footprints. The advantages are

Alessandro Antonucci, Paolo Bevilacqua, Stefano Leonardi, Luigi Paolopoli and Daniele Fontanelli have contributed equally to this work.

✉ Daniele Fontanelli
daniele.fontanelli@unitn.it

Alessandro Antonucci
alessandro.antonucci@ex-staff.unitn.it

Paolo Bevilacqua
paolo.bevilacqua@ex-staff.unitn.it

Stefano Leonardi
stefano.leonardi@unitn.it

Luigi Paolopoli
luigi.palopoli@unitn.it

[1] Department of Information Engineering and Computer Science (DISI), University of Trento, Via Sommarive 9, 38123 Trento, Italy

[2] Department of Industrial Engineering (DII), University of Trento, Via Sommarive 9, 38123 Trento, Italy

manyfold. From the robot's perspective, the human acts as an external module for the motion planning task, simplifying the complexity of the software components and of the sensing systems, while enabling the motion in a-priori unknown environments. From the perspective of the operator, s/he is in condition to drive a complex and heavy robot without any skill other than being able to walk and to have an elementary understanding of the robot's motion constraints (e.g., the robot cannot fly or using stairs). The robot operates semi-autonomously, i.e. it does not interfere with the pathfinder choices nor does it modify the path. However, it is allowed to stop when an obstacle materialises below a safety distance.

Therefore, our system is required to comply with two requirements:

Q1: The robot shall follow the path-finder even if s/he falls outside of the visual cone of the camera: the path has to be reconstructed and exactly followed even after sharp turns. This marks a remarkable departure from standard visual servoing approaches, which require the human to constantly remain within the robot's field of view.

Q2: The robot shall not collide with humans and obstacles. Although the path-finder is assumed to follow a safe path, the robot has to react to the unpredicted changes typical of a dynamic environment.

Our processing and execution pipeline has three phases: 1. Identification of the path-finder within the front camera image frames, 2. Fusion of the visual information with the one coming from other sensors, 3. Reconstruction of a smooth and feasible path from the time series of the path-finder's positions to be followed by a controlled motion along the path. The first phase is troublesome because the path-finder position is extracted from a noisy source, in which an ambiguous classification of the different subjects is quite frequent. Our solution is to split the first phase in three sub-phases. The first one detects the objects of interest within the image using a state-of-the art Convolutional Neural Networks (CNN) detector. The second sub-phase recognises the path-finder between the objects detected in the image. The feature identification is kick-started during the starting phase and is continuously refined during the system operation. The recognition properly said is performed by a K-Nearest Neighbour (KNN) classifier. The third sub-phase consists of a tracking module, which ensures continuity in the estimated positions of the target across different frames. In the second phase, the image information is fused with the measurements of a LIDAR sensor to reconstruct the correct location of the target and its headway distance from the robot. The third phase processes the time series of the estimated position of the path-finder, refining the path and guiding the navigation. This step uses clothoid curves to interpolate the points, which produces a path with continuous curvature and easy to follow for a robot. Finally, the control module follows the estimated path and enforces the necessary safety policies.

1.1 Summary of the paper contribution

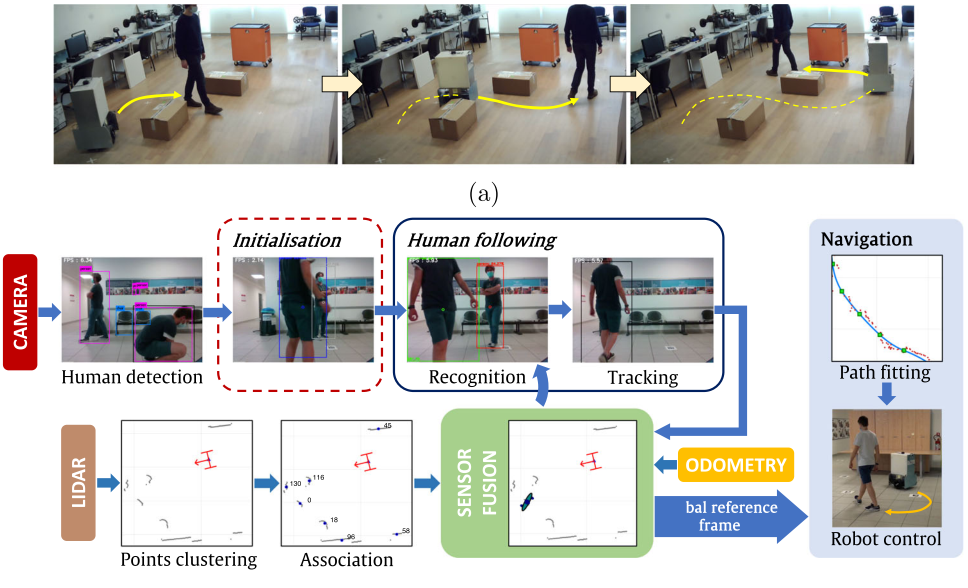
The idea outlined above can be seen as an original and modern application of the teach-by-showing approach to mobile robots moving in a complex scenario. This is classified in the recent literature (Islam et al. 2019) as a very relevant and largely open problem and is the key methodological contribution made in this paper. As a result, planning in environments that are a-priori unknown to the robot becomes feasible, which is a remarkable novelty for the field. Other two contributions have a more technical nature and descend from the complexity of our safety and reliability requirements. The first of them is the combination of tracking filter and neural network to estimate and follow the path-finder's position, which allows us to follow the path-finder even when s/he falls outside of the camera's visual cone. The second one is the idea to feedback the fused estimate into the recognition module and to exploit a trained neural network using its last layer to classify the person's feature set. This solution significantly improves the system's ability to distinguish between persons with similar features and resolve misclassifications due to illumination changes and partial occlusions (see Fig. 1-b).

1.2 Paper organisation

The paper is organised as follows. In Sect. 2, we summarise the most important existing results that we used as reference for this work. In Sect. 3, we present our general architecture and provide details on the perception components. In Sect. 4, we show our solution for path reconstruction and the control strategy for following the path. The experiments supporting the validity of the approach are described in Sect. 5. Finally, in Sect. 6 we give our conclusions and announce future work directions.

2 Related work

People following is a complex activity requiring a combination of perception, planning, control, and interaction strategies. Following a specific person rather than any person adds more to the complexity of the problem and is largely classified as an open problem. The main issue is that in a complex scenario many people can look similar if they do not wear specific markers. Most of the methods developed in the last decade and surveyed by Islam et al. (2019) claim a good performance in detection and tracking of humans but none of the papers cited in the survey covers the requirements of the application presented in this paper. Target re-identification and recovery has been obtained in the literature by using probabilistic models (e.g., Kalman filters) (Zhou et al. 2008), feature-based techniques (Layne et al. 2012) and, more recently, appearance-based



(a)

(b)

Fig. 1 **a** Different snapshots representing different time instants of the robot following the human operator across an environment with various static obstacles. **b** Flow diagram of our framework. The system starts with the initialisation procedure, collecting visual features of the path-finder. Then, in the path-finder following phase, the recognition module retrieves the new path-finder's position (see Sect. 3.2). The sensor fusion module fuses the camera tracking with the data from the LIDAR sensor, and redirects back the information to the recognition module (see Sect. 3.3). Finally, the set of the path-finder's positions over time are forwarded to the path reconstruction module (see Sect. 4.1) and the control module (see Sect. 4.2)

deep networks (Quispe and Pedrini 2019). However, the combination of these methods within a robotic application have not been investigated. Specifically, human-following applications require the combination of sophisticated learning approaches, model based filtering and path interpolation, as shown in this paper. The large majority of the solutions presented in Islam et al. (2019) does not perform re-identification and thus are not suitable to be applied in populated environments as ours. Only a small minori ty of the surveyed papers (Eisenbach et al. 2015; Koide and Miura 2016; Chen et al. 2017) propose target re-identification, but none of them considers explicitly robot planning and control. Even when the authors propose a human following approach (Gupta et al. 2017; Germa et al. 2010), they implement a visual servoing controller using a camera with a limited field-of-view. On the contrary, the point of this paper is using a human path-finder to identify the correct path, which requires accurate path reconstruction and path planning. In the entire survey, the only paper proposing an idea somewhat similar to our own is Doisy et al. (2012): here the authors use an RGB-D camera mounted on a turning platform for continuous person tracking. However the authors make the assumption that the only human in the scene is the target and that obstacles have a low height compared to the mounting point of the camera. On the contrary, the solution proposed in this paper does not make assumptions on the obstacles, and, as clearly visible from our experiments, it can be safely used in crowded environments.

2.1 People following

The combination of detection, tracking, and recognition was proposed by Jiang et al. (2018) using Speeded Up Robust Features (SURF). Chen et al. (2017) employed an adaptive boosting (AdaBoost) together with a stereo camera to real-time track a person, where the depth information is used to reinforce the classifier. Their approach can deal with appearance changes, people with similar clothes, and complete occlusions, but follows a classic visual-servoing approach: the robot control module is programmed to keep the target always within the camera frame, that is a remarkable difference with respect to our approach. Similarly, Wang et al. (2018) combined a monocular camera with an ultrasonic sensor to fuse range information with Kernelised Correlation Filters (KCF) based visual tracking. Their system has been tested in the case of visual interferences such as occlusions and illumination changes, however, due to the nature of the sensors employed, the human must remain in the camera view, and there is no specific strategy if the human's appearance changes. An implementation of RGB-D camera, laser scanner, and EKF is used by Nikdel et al. (2018) for their following-ahead mobile platform. Their framework likewise assumes that the human will often be outside the camera view, so the laser data and a non-holonomic human motion model are used to recover missing image data. Nevertheless, the presence of multiple humans undermines the tracking performance, which is instead one of the positive features of our solutions.

2.2 Background material on vision-based techniques

2.2.1 Object detection

Object detection is in our framework the first element of the processing pipeline. For this component, we sought a good compromise between classification accuracy and achievable frame rate. The available methods range from object detection and segmentation methods (Liu et al. 2016; Redmon et al. 2016; Girshick et al. 2014), to specific solutions for human pose detection such as OpenPose (Cao et al. 2019). YOLO (Redmon et al. 2016) is a very effective solution based on a single CNN; its main known disadvantage materialises when two classes have similar probabilities or the shape of the element is not perfect and the algorithm could produce different bounding boxes for the same object. Alternative solutions such as SSD (Liu et al. 2016) apply correction techniques to overcome the limitation of the approach (Neubeck and Van Gool 2006). SSD is also based on a single CNN to produce bounding boxes, but internally performs Non-Maximum Suppression (NMS) to remove unnecessary detection. Moreover, while the architecture of YOLO is designed as a compact block, SSD is instead modular, divided into convolution layers of different scales combined at the end.

2.2.2 People recognition

People recognition in computer vision is difficult in its own right. An additional level of complexity for robotics applications is introduced by the fact that the camera used for image acquisition is mobile. Traditional offline algorithms like Support Vector Machines (SVM) (Hearst et al. 1998) are known to react quickly to classification queries, but are not a good fit for our scenario, because we lack a prior knowledge on who is going to be the path-finder and we need to be robust against possible changes in her/his appearance. Methods based on key feature point matching (Pun et al. 2015) are known to be robust and are widely used to find small patterns in complex images, but in our tests the PRID450 (Person Re-IDentification) dataset (Roth et al. 2014) showed a high number of errors for low-res images and for deformable shapes such as humans clothes (see Sect. 5.1). Our final solution was based on the use of a K-Nearest Neighbours (KNN) classifier, which is an efficient training-free classification method albeit it requires the knowledge of representative points for the classification. For this information we used the last layer of a CNN, which gets trained with the different views of the path-finder. The idea of using a CNN classifier to extract the feature set was presented by Ristani and Tomasi (2018), who proposed a solution to match detections from multiple cameras. The classifiers evaluated for comparison in this work are the Deep Neural Networks (DNNs) based GoogLeNet (Szegedy et al. 2015) and ResNet (He et al. 2016). ResNet architecture is made of convolution blocks stacked one after the other, with an additional identity connection that preserve the input image through several layers of the network. GoogLeNet introduced the so-called inception module, which parallelises three different convolution filters and a max-pooling filter.

2.2.3 Person tracking

For person tracking, we could select from a large variety of approaches for the tracking of general objects (the fact that our object of interest is a person does not make a big difference in this case). Specifically, we considered: the Multiple Instance Learning (MIL) tracker (Babenko et al. 2010), the Kernelised Correlation Filters (KCF) tracker (Henriques et al. 2012), the Median Flow tracker (Kalal et al. 2010), the Channel and Spatial Reliability Tracker (CSRT) (Lukezic et al. 2017), the Minimum Output Sum of Squared Error (MOSSE) tracker (Bolme et al. 2010), the Generic Object Tracking Using Regression Networks (GOTURN) tracker (Held et al. 2016), and the Tracking-Learning-Detection (TLD) (Kalal et al. 2011). The MIL tracker (Babenko et al. 2010) is trained online during the execution of the tracking by generating negative samples from bounding boxes that do not overlap the correct one and by creating multiple instances around the correct sample for classification improvement. The KCF tracker (Henriques et al. 2012) is an extension of the MIL tracker which relies on Fast Fourier Transformations to increase accuracy and speed, but its weakness stands in full occlusions. The Median Flow tracker (Kalal et al. 2010) is a reliable method that locates the subject according to its trajectory, thus using an estimation of its motion model, however, despite its being robust, it suffers with high deformable subjects such as animals or humans. The CSRT (Lukezic et al. 2017) uses an high number of cross correlation filters in order to reach a very high accuracy, compensated by a low frame-per-second (FPS) rate. The authors of the MOSSE tracker (Bolme et al. 2010), based on the MOSSE correlation filter, state robustness against variations in lighting, scale and non-rigid deformations, moreover, in our experiments it showed extremely fast computations, i.e. high FPS rate (see Sect. 5.1). The GOTURN tracker (Held et al. 2016) is based on an offline trained CNN, hence can perform at very high FPS rate. However, since it takes one frame at a time and always compares it to the previous one, this algorithm suffers total occlusions. Differently from all the other methods, the TLD (Kalal et al. 2011) is able to overcome long-time total occlusion and to offer a long-term tracking, which are paid by a low FPS rate and a huge quantity of false-positive predictions.

2.3 Sensor fusion

Our application requires 3D reconstruction of the human pose. The combination of stereo and RGB-D sensor with skeleton-based approaches proves very useful to this purpose and it is significantly simplified by the availability of public domain software components (Antonucci et al. 2019). However, the simple use of visual information has known limitations, such as the sensitivity to lighting conditions and the high computation times. Laser-based sensors, on the other hand, are relatively reliable on a long range and are less computation hungry than vision based approaches. However, recognising a specific person from a slice of a 2D point cloud is hopeless. For this reason, moving along a direction frequently taken in robotics (Zhen et al. 2019; Wolcott and Eustice 2014; Nguyen et al. 2021), we apply a combination of cameras and LIDARs. The use of separate systems for depth estimation and classification improves the robustness of the tracking system when one of the sensors fails: if the human falls outside the camera field of view, we keep using the LIDAR sensor for tracking, whose reliability changes according to the adopted human motion model.

3 Tracking the human path-finder

Before going into the details of the algorithm we use to track the human, we succinctly describe the available sensing system and the model of the platform. The reference model for the robot is in this paper the unicycle, whose kinematics can be described in discrete–time by the following Zero-Order-Hold model:

$$\mathbf{s}(t_{k+1}) = \begin{bmatrix} x_r(t_k) + \cos(\varphi_r(t_k))(t_{k+1} - t_k)v_r(t_k) \\ y_r(t_k) + \sin(\varphi_r(t_k))(t_{k+1} - t_k)v_r(t_k) \\ \varphi_r(t_k) + (t_{k+1} - t_k)\omega_r(t_k) \end{bmatrix} \qquad (1)$$

where $\mathbf{s}(t_k) = [x_r(t_k), y_r(t_k), \varphi_r(t_k)]^T$ is the state of the robot, the Cartesian coordinates $(x_r(t_k), y_r(t_k))$ refer to the mid-point of the rear wheels axle in the $X_w \times Y_w$ plane expressed in the $\langle W \rangle = \{X_w, Y_w, Z_w\}$ world reference frame, $\varphi_r(t_k)$ the longitudinal direction of the vehicle with respect to the X_w axis, $v_r(t_k)$ and $\omega_r(t_k)$ the longitudinal and angular velocities, respectively, and t_k the reference time instant, which is usually chosen as an integer multiple of a fixed sampling time. Importantly, the proposed framework would be applicable to different robot dynamics; however, as explained next, the unicycle structure is particularly convenient for the class of applications we address.

Without loss of generality, we assume here that the choice of the sampling time $\delta_t = t_{k+1} - t_k$ is imposed by the sensor with the lowest sampling frequency. The assumed sensing configuration is based on the presence of rotation encoders on each of the rear wheels or any other sensing system able to provide ego-motion informations (e.g., IMUs, visual odometry). For the perception of the surroundings, the sensing system comprises a LIDAR and an RGB-D camera. The LIDAR data are used to both track humans around the vehicle and to localise the vehicle inside the environment, using a standard approach presented in the literature (Hess et al. 2016). The RGB-D camera is primarily used for the human detection and tracking. The laser scanner (an RPLIDAR A3[1]) employed has a view of 360°, a maximum measuring distance up to 40 ms, and is typically operated at 20 revolutions per second. The RGB-D camera adopted is an Intel® RealSense™ D435,[2] working in an ideal range spanning from 0.5 to 3 m, whose produced data are used in the vision-based detection and recognition system described in Sect. 3.2. The LIDAR and the camera are rigidly mounted on the top of the robot chassis (see Fig. 3a) and return the measurements at time t_k in the LIDAR $\langle L_k \rangle$ and camera $\langle C_k \rangle$ reference frames, respectively, which are bot rigidly linked to the robot (i.e., they operate with a local coordinates reference system). The transformation matrix ${}^L T_C$ between the two frames is estimated during an initial calibration phase.

3.1 Solution overview

The proposed scheme is sketched in Fig. 2. A first group of processing activities operates in the local frame, where it seeks to detect and track the path-finder. Such activities are based on two distinct and converging flows of information. The first flow (Vision-based detection and recognition) comes from the RGB-D camera and allows us to identify and track the path-finder position within the image space. The second comes from the LIDAR sensor and looks for the same information from a different source with three different purposes. The first purpose is to increase the robustness of the vision based tracking by injecting the LIDAR data into the recognition activity. The second is to improve the accuracy of the estimation by fusing depth and visual information. The third is to allow the system to track the path-finder even when s/he falls out of the RGB-D sensor visual cone. A second group of processing activities takes place in the global reference frame. The main objective is to transform the time sequence of the path-finder's positions into a set of smooth

[1] https://www.slamtec.com/en/Lidar/A3.

[2] https://www.intelrealsense.com/depth-camera-d435/.

Fig. 2 Overall scheme of the algorithm

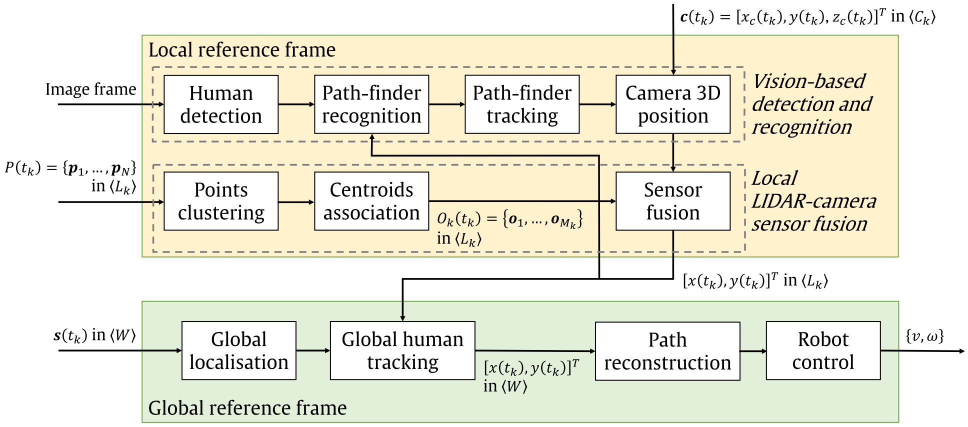
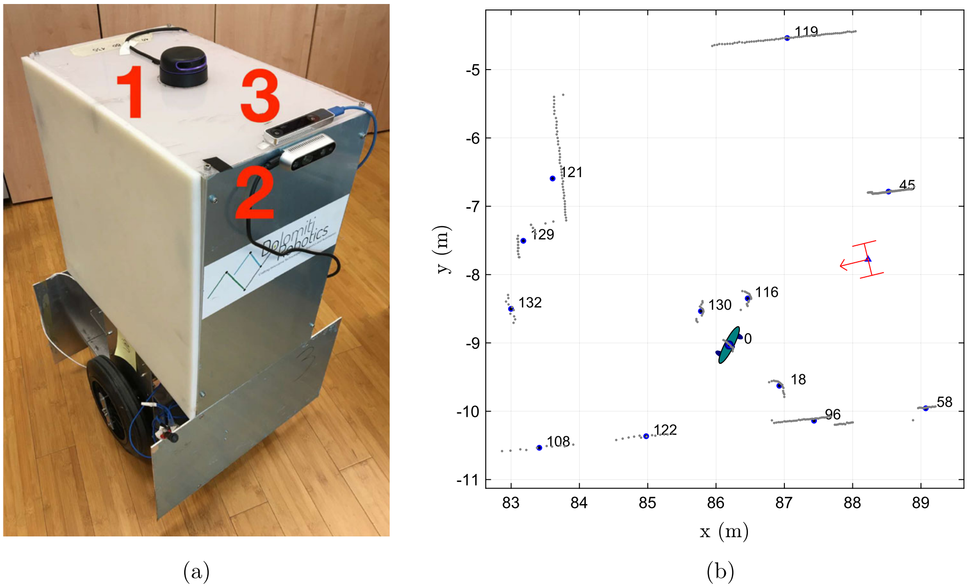

$c(t_k) = [x_c(t_k), y(t_k), z_c(t_k)]^T$ in $\langle C_k \rangle$
Local reference frame
Image frame
Human detection
Path-finder recognition
Path-finder tracking
Camera 3D position
Vision-based detection and recognition
$P(t_k) = \{p_1, \ldots, p_N\}$ in $\langle L_k \rangle$
Points clustering
Centroids association
$O_k(t_k) = \{o_1, \ldots, o_{M_k}\}$ in $\langle L_k \rangle$
Sensor fusion
Local LIDAR-camera sensor fusion
$[x(t_k), y(t_k)]^T$ in $\langle L_k \rangle$
$s(t_k)$ in $\langle W \rangle$
Global localisation
Global human tracking
$[x(t_k), y(t_k)]^T$ in $\langle W \rangle$
Path reconstruction
Robot control
$\{v, \omega\}$
Global reference frame

Fig. 3 **a** Robot sensing system setup, consisting of LIDAR sensor (1), RealSense D435 (2) for the path-finder detection and tracking, and RealSense T265 (3) for the visual odometry. **b** Representation of 2D data measured by the robot (red), with the raw laser scanned cloud points (thin grey dots) and their corresponding object centroids (thick blue points) along with the IDs (number in the figure), expressed in the LIDAR reference system $\langle L_k \rangle$ at time t_k. The human position (green shape with ID 0) can be retrieved with the sensor fusion (Color figure online)



(a)

(b)

geometric motion primitives in order to have them followed by the robot.

3.2 Vision-based detection and recognition

The vision based algorithm takes the lion share in our tracking solution, but, as we mentioned above and as explained in Sect. 3.3, the tracking performance and robustness is significantly improved by the integration of the LIDAR data. In the next paragraphs we will first describe the different components and then discuss how they interoperate in an integrated pipeline.

3.2.1 Human detection

The first activity of the detection and recognition algorithm is to localise D people inside an image frame. To this end, we have chosen the latest version of YOLO available, YOLOv3 (Redmon and Farhadi 2018), and a lighter implementation of SDD, namely MobileNet (Howard et al. 2017) (designed to execute on low power devices). The detection module identifies the objects in view through the smallest bounding box that contains the required element. In the starting phase, the person associated with the largest bounding box is recognised as the path-finder.

3.2.2 Path-finder recognition

This module is used to understand which of the humans found in the frame corresponds to the path-finder, thus also enabling a coherent connection between the detector (based on a KNN classifier) and the tracker. The KNN classifier uses the vector points generated by two DNN image classifiers: ResNet50 (He et al. 2016), which produces a representation point in 2048 dimensions, and GoogLeNet (Szegedy et al. 2015), which produces a representation point in 1024 dimensions. If the path-finder is contained in the list of the D people detected, the information is passed to the image tracker, otherwise the procedure loops back to the detection phase. As previously mentioned, this phase also exploits the data coming from the sensor fusion phase in order to improve the tracking performance. This important feature is described in detail below. If the estimation error of the global human tracking (introduced in Sect. 3.4) exceeds the desired path-finder tracking uncertainty due to repetitive sensor or detection failures, the system reaches a faulty condition, the robot stops and the process restarts from scratch.

3.2.3 Path-finder tracking

This module is periodically executed to track the path-finder location. The pathfinder is sought with a number of frames chosen as a fixed parameter m in order to avoid the problems associated with long-term sequences. We implemented for the tracker the method that best fitted our requirements, i.e. KCF, CSRT and MOSSE, and we finally adopted the KCF in our experiments, since it offers a good compromise between robustness and speed. We emphasise that if a single detection fails or the path-finder is not found, the tracker cannot return a reliable measurement; however, the sensor fusion module can recover the correct path-finder location thanks to the position evaluated at the previous iteration.

3.2.4 The vision pipeline as a whole

The system operates in two phases: *initialisation* and *human following*. The initialisation phase is needed because our vision processing pipeline leverages a learning-based classification of the human pathfinder, which in turn requires the knowledge of her/his features. During this phase, which lasts for Δ_t seconds, the robot collects a series of bounding boxes used to create the set of positive representative points into the N-dimensional space of the KNN. Simultaneously, a negative sample is randomly picked up from a database and it is also given to the KNN to balance the number of positive and negative samples. The negative samples come from our customised version of the Market1501 dataset (Zheng et al. 2015). An example of the initialisation phase is shown in Fig. 4a.

When the system switches to the *human following* phase, it carries out a first detection step. Then, it uses a KNN classifier in order to distinguish between positive detections and negative ones. In order to make the classification robust, the system uses the information from the path-finder position estimated at the previous time step (see Sect. 3.3). If the Euclidean distance between the last estimated position and the 3D position measured by the camera is lower than a threshold d_f, the detected position is considered as valid and the new set of visual features is added to the positive dataset. Detections without the described distance correspondence can be added to the negative dataset. This simple feedback mechanism significantly improves the system reliability and its resilience to wrong classifications or changes in the path-finder's appearance. Finally, the path-finder tracking module is executed, and the resulting information is passed to the fusion module. After the successive m frames, the detection and recognition stages are re-executed in order to strengthen the tracking performance.

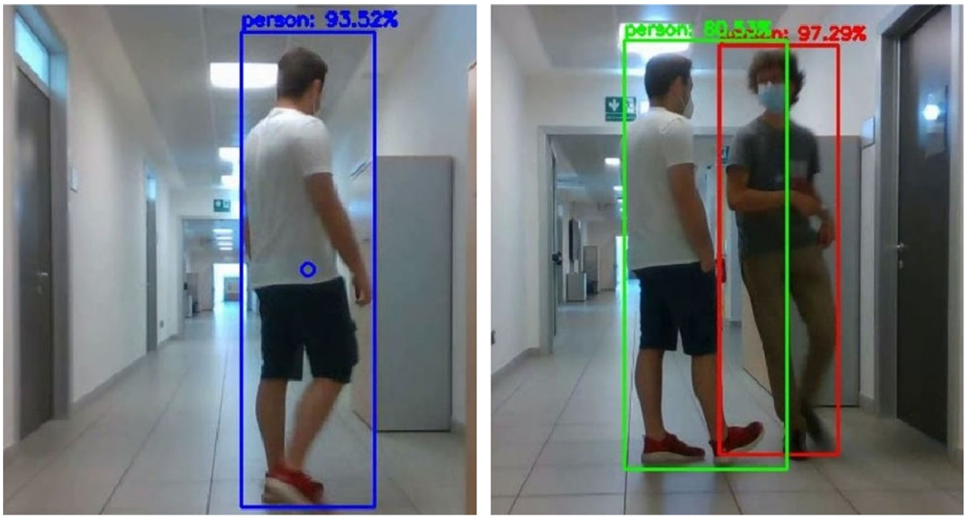
This processing workflow and the different feedback cycles it is based on delivers a highly performant image processing and an improved robot localisation accuracy, as shown by the experimental data in Sect. 5.

3.3 Local LIDAR-camera sensor fusion

The information coming from the vision-based algorithm are combined with the LIDAR in the local reference frame $\langle L_k \rangle$ in order to make the procedure more robust, as aforementioned. Moreover, the path-finder can be tracked for some time also when s/he evades the vision cone of the RGB-D camera just relying on the LIDAR information.

3.3.1 LIDAR clustering

The sensor reading delivered by the laser scanner at time t_k provides a sequence of N_k measurement points in the form of $\mathcal{P}_k = \{\mathbf{p}_1, \dots, \mathbf{p}_{N_k}\}$, represented in polar coordinates as $\mathbf{p}_i = (r_i, \alpha_i)$, i.e. the range r_i and the angle α_i expressed in the planar LIDAR reference frame $\langle L_k \rangle$ (see Fig. 3b for an example of an actual scan). At time t_k, the measured points are filtered and grouped into M_k clusters based on the mutual Euclidean distances and on the richness, i.e. on a minimum number of sensed points for each cluster, each identified by the object centroid $\mathbf{o}_j(t_k) = [x_j(t_k), y_j(t_k), 0]^T$, $j = 1, \dots, M_k$ (see the ID numbers in Fig. 3b and the Algorithm 1). Given two sets of objects $\mathcal{O}_k = \{\mathbf{o}_1(t_k), \dots, \mathbf{o}_{M_k}(t_k)\}$ and $\mathcal{O}_{k+1}$, taken in two consecutive time instants t_k and t_{k+1} and possibly having $M_k \neq M_{k+1}$, we adopt the Munkres assignment algorithm (Munkres 1957) to decide either if the two objects are actually the same or if a new object has been detected. To this end, since between t_k and t_{k+1} the robot moves for δ_t seconds according to the model (1), we can update its position $\mathbf{s}(t_{k+1})$ in $\langle L_{k+1} \rangle$ either by using ego-motion data or, if available, the global localisation module. After the motion, $\mathcal{O}_k$ previously expressed in $\langle L_k \rangle$, is projected in new local frame $\langle L_{k+1} \rangle$. The presence of LIDAR measurement noise imposes the use of a forgetting factor, hence the algorithm removes the object whose centroid does not find correspondences for T_fd time instants. This way we can disambiguate the different entities in the robot's surroundings, increase robustness to partial occlusions and exploit indeed a prior-based tracking (since each cluster has its own unique ID, as depicted in Fig. 3-b).



(a) (b)

Fig. 4 **a** Initialisation phase: the detected path-finder is depicted with a blue rectangle. **b** Following phase: the path-finder ISO correctly recognised (green rectangle), while another person is a negative sample (red rectangle) (Color figure online)

Algorithm 1 Laser scanner points clustering algorithm

```
Input: $\mathbf{p}_i = (x_i, y_i), i = 1, \dots, N$ // Get measurement points
$j = 1, \mathbf{c}_j \leftarrow$ Append($\mathbf{p}_1$) // Initialisation first object
for $i = 2$ to $N$ do
    if $||\mathbf{p}_i - \mathbf{p}_{i-1}||_2 < d_c$ then
        $\mathbf{c}_j \leftarrow$ Append($\mathbf{p}_i$) // Add $\mathbf{p}_i$ to the set $\mathbf{c}_j$
    else
        if $\dim(\mathbf{c}_j) > m_c$ then
            $\mathbf{o}_j = \sum_{k=1}^{\dim(\mathbf{c}_j)} \mathbf{c}_j(k)/\dim(\mathbf{c}_j)$ // Object centroid
        end if
        $j \leftarrow j + 1, \mathbf{c}_j \leftarrow$ Append($\mathbf{p}_i$) // Start new object
    end if
end for
Output: $\mathcal{O} = \{\mathbf{o}_1(t_k), \dots, \mathbf{o}_j(t_k)\}$
```

3.3.2 Camera 3D position

The tracking module described in Sect. 3.2 returns a bounding box $[x, y, w, h]$ in the image frame, containing the (x, y) pixel coordinates of the top-left corner of the box, its width w and height h, which is then converted in the $\langle C \rangle = \{X_c, Y_c, Z_c\}$ pin-hole camera reference system. Notice that the depth information along the Z_c axis is retrieved via the RealSense™API. As a consequence, the centroid of the i-th bounding box $\mathbf{c}_i(t_k) = [x_c(t_k), y_c(t_k), z_c(t_k)]^T$ can be expressed in the camera reference system $\langle C_k \rangle$ at time t_k.

3.3.3 Sensor fusion

Given the set of objects $\mathcal{O}_k$ and the centroid(s) of the bounding box(es) $\mathbf{c}_i(t_k)$, taken at the same time instant t_k, we adopt a spatio-temporal correspondence algorithm with the two sets of measurements to decide if the tracked object is the same or a new one has entered into the scene. Our algorithm is implemented as a finite state machine, comprising the *Init* and *Track* states (see Algorithm 2). The rationale is the following: in the *Init* state, we look for a correct match between the j-th clustered object $\mathbf{o}_j(t_k)$ and the i-th bounding box centroid $\mathbf{c}_i(t_k)$, which occurs when their Euclidean distance in the local LIDAR frame $\langle L_k \rangle$ is below a threshold d_p (this is obtained in $\langle L_0 \rangle$ at the end of the *initialisation phase*, where the path-finder stands in front of the robot for the initial bootstrap). When the correct match is found with the same j-th object $\mathbf{o}_j$ for m_p time instants, the j-th object is "promoted" as PATH- FINDER $\mathbf{o}^\star_j(t_k)$, while all the other objects are labelled as OBSTACLE, whereupon the state changes to *Track*. Notice that this procedure reduces at the same time the computation times and the probability of mismatch, while making the algorithm robust to sensor failures (either the bounding box or the LIDAR cluster are sufficient for recognition). In the *Track* state, at time t_{k+1}, the match is evaluated for the $\mathbf{o}^\star_i(t_{k+1})$ only, as all the possible new objects in $\mathcal{O}_{k+1}$, by default labelled as UNKNOWN, become OBSTACLE objects. When a mismatch between the $\mathbf{o}^\star_j(t_{k+1})$ and $c_i(t_{k+1})$ occurs for m_r time instants, the track is rejected, i.e. all objects become UNKNOWN again, and the algorithm switches back to the *Init* state. Possible mismatch events happen if the Euclidean distance in $\langle L_{k+1} \rangle$ is higher than d_p or the object $\mathbf{o}^\star_j(t_{k+1})$ itself is removed during the clustering association. Finally, the Cartesian coordinates of the path-finder $(x(t_{k+1}), y(t_{k+1}))$ in the local frame $\langle L_{k+1} \rangle$ are propagated back to the people recognition module (see Sect. 3.2) to strengthen the human tracking, since such information form a prior for the next path-finder detection.

3.4 Global human tracking

Since the human is used as a path-finder for future executions of the path, her/his position should be estimated in the global

Algorithm 2 Sensor fusion algorithm

```
Input: $\mathcal{O}(t_{k-1})$ in $\langle L_{k-1}\rangle$, $\mathcal{O}'(t_k)$ in $\langle L_k\rangle$, $\mathbf{c}_i(t_k)$, $\mathbf{s}(t_k)$
$\mathcal{O}(t_{k-1})$ in $\langle L_k\rangle \leftarrow$ RotoTranslate $(\mathcal{O}(t_{k-1}), \mathbf{s}(t_k))$
$\mathcal{O}(t_k) \leftarrow$ Munkres $\left(\mathcal{O}(t_{k-1}), \mathcal{O}'(t_k)\right)$
if State == $Init$ then
    $\mathbf{d}_j = ||\mathbf{o}_j(t_k) - \mathbf{c}_i(t_k)||_2, \forall j = 1, \dots, M_k$
    $j^* = \arg\min(\mathbf{d})$
    if $d_{j^*} < d_{\mathrm{p}}$ then
        $pm_{\mathbf{o}_j^*} \leftarrow pm_{\mathbf{o}_j^*} + 1$ // Number of matches
    end if
    if $\exists\, \mathbf{o}_j \in \mathcal{O} \mid pm_{\mathbf{o}_j} > m_{\mathrm{p}}$ then
        $\mathbf{o}_j \leftarrow \mathbf{o}_j^\star$
        $\mathbf{o}_k \leftarrow \mathbf{o}_j^{\mathrm{ob}}, \forall k \neq j$
        State $\leftarrow Track$
    end if
end if
if State == $Track$ then
    $d = ||\mathbf{o}_j^\star(t_k) - \mathbf{c}_i(t_k)||_2$
    if $d > d_{\mathrm{p}}$ then
        $fm_{\mathbf{o}_j^*} \leftarrow fm_{\mathbf{o}_j^*} + 1$ // Number of mismatches
    end if
    if $fm_{\mathbf{o}_j^\star} > m_{\mathrm{r}}$ then
        $\mathbf{o}_j \leftarrow \mathbf{o}_j^{\mathrm{un}}, \forall j = 1, \dots, M_k$
        State $\leftarrow Init$
    end if
end if
Output: $(x(t_k), y(t_k))$ in $\langle L_k\rangle$
```

reference frame $\langle W\rangle$. To this end, we first need to estimate the robot position $s(t_k)$ in $\langle W\rangle$. This is accomplished fusing together the encoder readings, the visual odometry and the LIDAR points $p_i(t_k)$ with an a-priori map of the environment (if available) or solving a SLAM problem. The $s(t_k)$ robot position and the path-finder local measurements in $\langle L_k\rangle$ are used to obtain the Cartesian coordinates $(x(t_k), y(t_k))$ of the path-finder in $\langle W\rangle$.

To track the human in $\langle W\rangle$, an estimation algorithm is needed, whose main role is to further improve the accuracy of the reconstructed path and to further increase the robustness to occasional sensor failures. In order to limit the computational cost, we make the assumption that the path-finder moves with a velocity following a Gaussian probability density function. This random walk hypothesis is quite standard in the literature and derives from the lack of knowledge about the actual human motion intentions. What is more, following the observation that humans actually move with a smooth dynamic (Arechavaleta et al. 2008), the motion model can be approximated by a unicycle dynamic (Farina et al. 2017). Hence, we explicitly express the angular and linear velocities as states, and by denoting with $\bar{\mathbf{h}}(t_k) = [x(t_k), y(t_k), \theta(t_k), v(t_k), \omega(t_k)]^T$ the state at time t_k (where $v(t_k)$ and $\omega(t_k)$ are the forward and angular velocities, respectively), we have the following model

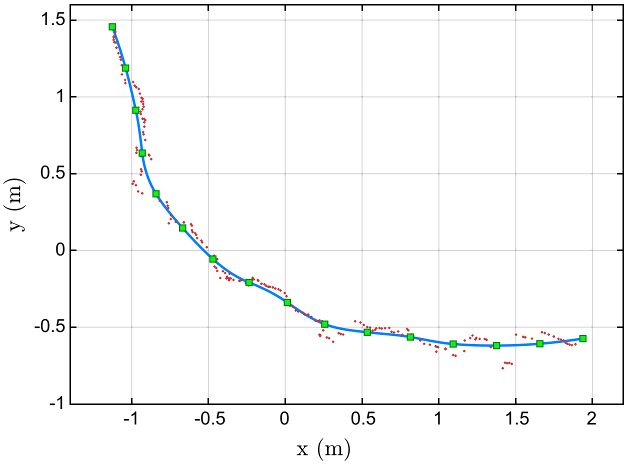


Fig. 5 Example of path fitting and reconstruction. The red stars represent the input data. The green squares are the fitted waypoints, sampled at a uniform distance along the path. The blue solid line is the reconstructed, smoothed path, to be followed by the robot (Color figure online)

$$\bar{\mathbf{h}}(t_{k+1}) = \begin{bmatrix} x(t_k) + \delta_t v(t_k)\cos(\theta(t_k)) \\ y(t_k) + \delta_t v(t_k)\sin(\theta(t_k)) \\ \theta(t_k) + \delta_t \omega(t_k) \\ v(t_k) \\ \omega(t_k) \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \delta_t & 0 \\ 0 & \delta_t \end{bmatrix} \begin{bmatrix} \eta_a(t_k) \\ \eta_\omega(t_k) \end{bmatrix} = \tag{2}$$
$$= f(\bar{\mathbf{h}}(t_k)) + B\eta(t_k),$$

where $\eta(t_k)$ is the acceleration noise affecting the linear and the angular velocities that is assumed to be $\eta(t_k) \sim \mathcal{N}(0, E)$, with E being its covariance matrix, and white (as customary). This model is easily used to generate the predictions in an Extended Kalman Filter (EKF) scheme.

4 Navigation

The aims of the navigation module are twofold: reconstructing the path followed by the path-finder in a form that can be followed by the robot, controlling the motion in order for the robot to follow the path with a good accuracy (small deviations are inevitable but they should be kept in check).

4.1 Path reconstruction

As shown in the scheme in Fig. 2, the path reconstruction module continuously receives new information on the current position of the path-finder from the perception module. This way, it creates a dataset composed of a time series of 2D path-finder positions, which are updated in real–time. The module executes a local path fitting of the estimated path-finder trajectory. An example execution of the process is shown in Fig. 5. The path is reconstructed using the following steps.

1. Once a new path-finder position is received, it is compared with the previous one, and, if the Euclidean distance is greater than a small threshold value, it is recorded into the dataset. This action is necessary to handle the scenario where the path-finder stops for a long time, in order to avoid an unnecessary growth of the dataset.
2. When the new position qualifies for its inclusion into the dataset, the x and y components of the data points are fitted using a classical smoothing algorithm, i.e. the LOESS (Locally Estimated Scatterplot Smoothing) (Cleveland 1979).
3. The fitted data points are then connected by a poly line, and a number of waypoints are sampled at a uniform curvilinear distance (corresponding to the green squares of Fig. 5).
4. The waypoints are connected by a G2 clothoid spline (corresponding to the solid blue line of Fig. 5), using the algorithms and techniques discussed in Bertolazzi and Frego (2018a), Bertolazzi and Frego (2018b), and for which an efficient C++ implementation is available (Bertolazzi et al. 2018).

The choice of the clothoid comes from the observation that humans tend to follow the unicycle-like dynamics (Farina et al. 2017) given in (2), which naturally generates clothoid curves. What is more, due to the continuity of the curves and of their curvature, clothoids have been proved to be effective to mimic a human path by a robotic agent (Bevilacqua et al. 2018).

4.2 Robot control

When a path is reconstructed, following the steps described above, the controller module takes the responsibility to execute a safe navigation of the robot following as closely as possible the prescribed path. For this work, we employed the path following algorithm described in Andreetto et al. (2017), which is velocity-independent and avoids the singularities presented by other common algorithms when the vehicle has to stop and the velocity is set to zero. The velocity of the robot is chosen by our controller based both on the distance from the end of the path (corresponding to the path-finder position), with the aim of following the path-finder at a constant (curvilinear) distance, on the current path curvature (the vehicle is slowed down when traveling a sharp curve), and on the past robot velocities (to limit the maximum allowed accelerations). Furthermore, the control module implements a safety policy whereby, when an obstacle is encountered along the path, the robot first slows down and then stops if the path remains occluded (notice that, since the path-finder have already travelled that area, the obstacle is necessarily dynamic, e.g. a human being, hence it is expected to pass by in a short time).

5 Experimental results

In this section, we first present our experimental evaluation to decide the most effective combination of solutions for vision-based detection and recognition. Then, we will report and discuss the performance of our system as a whole using a real robotic platform.

5.1 Vision module

We have organised the analysis of the vision module along three directions of prominent relevance for the application at hand: detection, recognition and tracking.

5.1.1 Detection

The comparison among the different possibilities aims to rate the computational efficiency (measured in FPS) and the algorithm precision (measured as mean Average Precision—mAP) from the data reported in the previously cited (Liu et al. 2016; Redmon et al. 2016; He et al. 2017; Redmon and Farhadi 2018; Cao et al. 2019). By observing the data, it appears that the single-stage algorithms (SSD and YOLO) are much faster with respect to the two-stage methods (R-CNN): they execute around 5–10 times faster than R-CNN. Instead, the precision of the three detectors is almost the same. Also from our experimental evaluation, SSD and YOLO perform better than R-CNN and OpenPose both in computational efficiency and algorithm precision. Our final choice fell on SSD, since it implements the CNN with a relatively small number of parameters: this ensures low execution time at the price of a slight detection inaccuracy (which is however compensated by the LIDAR data fusion, as discussed in Sect. 3).

5.1.2 Recognition

The first tests for the recognition module considered the feature point matching on the PRID450 (Person Re-IDentification) dataset (Roth et al. 2014), that contains thousands of cropped images of people walking outdoors. The dataset is constructed with multiple shots of the same person in different moments and perspectives. The results reported in Fig. 6 show the main problems of this techniques with human shapes. Humans present a highly deformable-body, with a surface (clothes) that continuously changes its aspect. However, the key points matching is designed for a pattern that is repeated often and clearly, as a consequence, the matching performance is not reliable at all. For example, even with the same subject with almost the same position (bottom-left couple of images in Fig. 6), the key points matching fails with most of the points. The exception is the bottom-right image that has a perfect matching, but the two pictures examined are exactly the same.

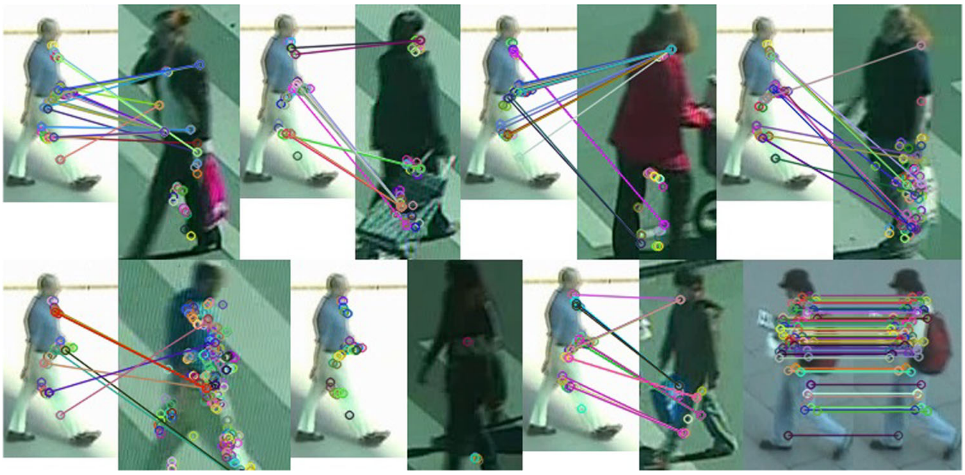


Fig. 6 Some example of matching samples with the key points matching on the PRID450 dataset. There are multiple failures: people who present no key points, objects such as bags that have plenty of features, matches that connect completely different parts of the body, like shoulders with legs

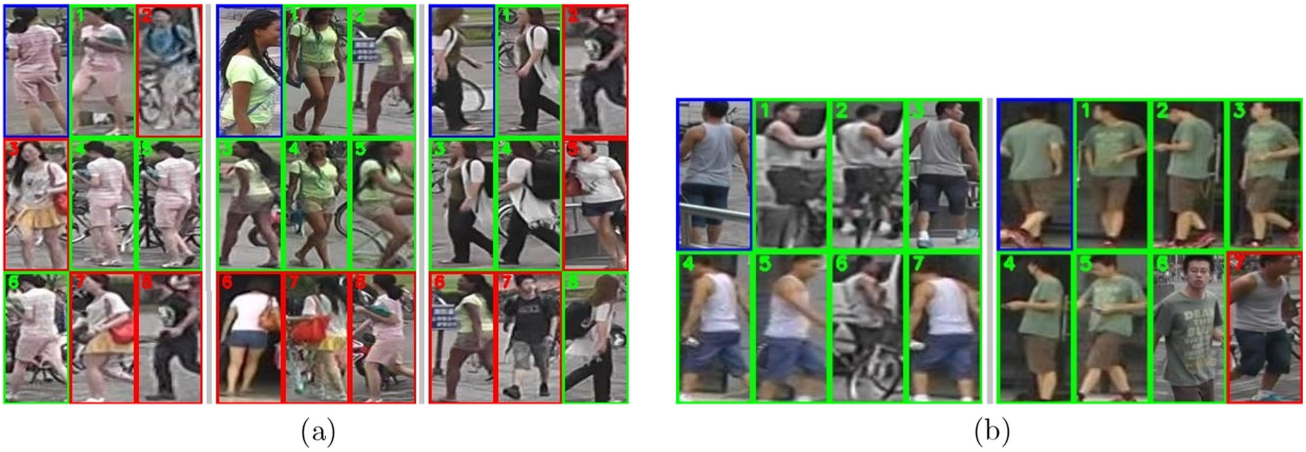


Fig. 7 KNN applied to images elaborated with 11 real people and ResNet50 (**a**) and with 2 real people and GoogLeNet (**b**), with 9 images of each person. The query (images with the blue contour) is used to extract from the database the most similar pre-analysed images. The green contour depicts the correct extracted person, whereas the red contour highlights wrongly extracted persons (Color figure online)

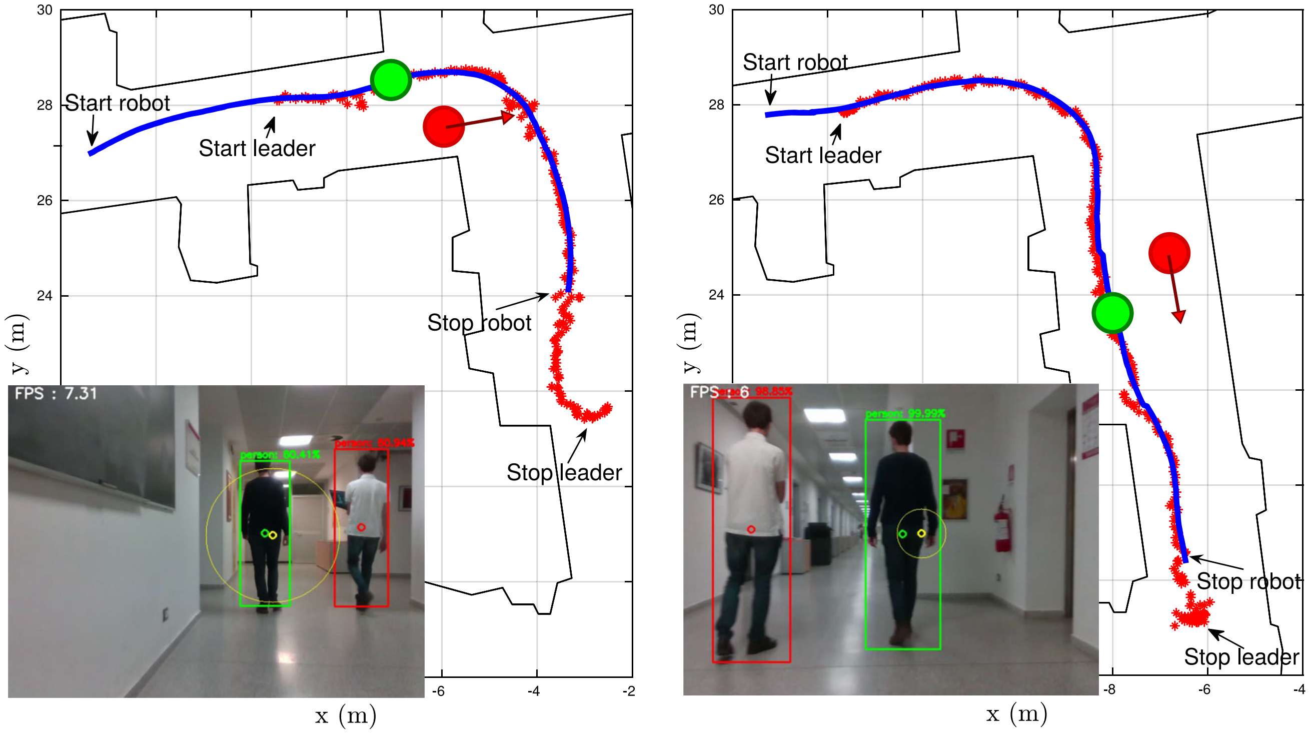
Instead, the proposed recognition module based on KNN has been also tested on the Market1501 dataset (Zheng et al. 2015) in conjunction with either ResNet50 or GoogLeNet. Similarly to the PRID450 dataset, the Market1501 contains sets of images of hundreds of people captured from different perspectives and in different moments. As an example, we selected two small datasets with 11 and 2 people respectively, with and 9 images per person, then we "trained" the KNN (i.e., we stored the data) with the representative points extracted from the images, and retrieved the most similar people. In Fig. 7, we report an example of the queries computed on the KNN classifier for the two chosen detectors. In the test elaborated with the first dataset, the classifier produced approximatively 50% of correct responses for each person (Fig. 7a). Instead, with 2 real human beings and 9 images per person, the KNN obtained only one false positive over 14 pictures. This result shows that the proposed KNN solution is appropriate for the application at hand (we are interested in only 2 classes, i.e., the path-finder and the other people). Similar results were obtained with ResNet50 and GoogLeNet, and we selected the second to work with the KNN since it achieved moderately faster computation times.

5.1.3 Tracking

For the image tracker, our aim is usually to process long real-time sequence with occasional total occlusions and changes of shape. Instead, since the complete camera pipeline is solved with a combination of detection, recognition, and tracking, the internal tracking task is simpler, and deals with changes of shape and partial occlusions. The requirement is to solve the task, and to deal with changes of shape and partial occlusions. The methods presented in Sect. 2.2 were evaluated in terms of execution time (for real-time implementation) and tracking performance.

Table 1 Overview of the frame-per-second (FPS) rate of the image tracking algorithms. The performance was measured on an Nvidia Jetson TX2 GPU

	MIL	KCF	MedFlow	CSRT	MOSSE	GOTURN	TLD
FPS	9	38	40	15	56	20	10



(a)

(b)

Fig. 8 Experimental trajectories in a hallway. The path-finder and a pedestrian walk in the same corridor with **a** partially occluding trajectories (EXP- 01) and **b** missing and recovering of the path-finder with the camera tracking (EXP- 02). Solid blue lines depict the trajectories followed by the robot, while the red stars mark the measured path-finder positions. The green and red circles correspond to the positions of the path-finder and the other pedestrian, respectively, when the inlet camera snapshots are grabbed (Color figure online)

Based on our results (shown in Table 1), the best trade-off choices from the application at hand were obtained with the KCF, CSRT, MOSSE, GOTURN trackers, and based on empirical evidence, we selected CSRT as the most suitable solution for our purposes.

5.2 Experiments with the mobile robot

The algorithms presented in Sect. 3 and Sect. 4 were executed on a Jetson TX2[3] for the acquisition of the RGB-D data and the classification, while the LIDAR scans, the sensing data fusion, and the navigation control were executed on a NUC, both on board of the wheeled robot entirely assembled at the University of Trento (see Fig. 3a). All the reported experiments were carried out in our department at the University of Trento. The source code for the described framework has been released and is publicly available.[4]

5.2.1 Reliability and robustness

In a first set of experiments, we aimed at testing the performance and robustness of the path-finder tracking algorithm. To this end, we recorded the real–time execution data in two different areas of an hallway with multiple exits and in different conditions. In Fig. 8a, the robot follows the path-finder while another pedestrian is walking nearby (EXP- 01). The algorithm successfully rejected the disturbing effect of the second pedestrian and it correctly followed the path-finder. Similar results (EXP- 02) were obtained for crossing trajectories or when the path-finder exits from the camera field of view for the right turn. Even in those cases, the other

[3] https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-tx2/

[4] https://github.com/AlexRookie/robotdog.

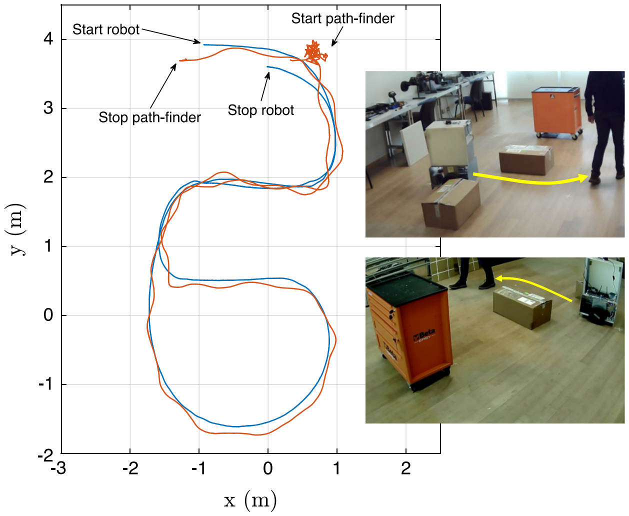


Fig. 9 An example of the actual path-finder path (red line) and robot trajectory (blue line) for the maze-like environment of EXP- 03 (Color figure online)

pedestrian is not wrongly classified as the path-finder, who is instead correctly tracked back after the turn (see Fig. 8b).

5.2.2 Accuracy

For a qualitative analysis of the tracking and navigation performance, we present in Fig. 9 the comparison between the robot trajectory (blue line) with the actual position of the path-finder (red line). Both trajectories were captured with a network of eight OptiTrack cameras for ground truth reference in a maze-like environment (EXP- 03). Notice the path-finder starting position standing in front of the robot during the bootstrap phase. The swinging path-finder trajectory is dictated by the OptiTrack tracked markers placed on the head of the human to avoid occlusions, hence oscillating with the footsteps. From this picture it is evident that, in sharp turns, the robot looses the image tracking of the path-finder, but it is nonetheless able to exactly follow the path by means of the fusion with the LIDAR data. Finally, we would like to point out that the error in the trajectory followed by the robot with respect to the human footsteps is in the range of ±25 cm, i.e. the typical encumbrance of the human body.

5.2.3 Robustness and accuracy for extreme manoeuvres hindered by an intruder

In Fig. 10 we report the experimental results in an hallway test (EXP- 04), where another human (i.e., a dynamic obstacle) moves in the scene. The path-finder was instructed to follow a winding path, which is a challenging scenario since the human necessarily and continuously exits from the field of view of the camera along the sharp turns. An additional element of complexity was given by the intruder (black dashed line) repeated interference with the robot operations. As visible in the plot, the path travelled by the robot (blue line) remained consistently aligned with the measured path-finder positions (red line). This experiments also shows the effectiveness of the LIDAR clustering and association algorithm presented in Sect. 3.3, which tracks the intruder and then rejects it for the evident inconsistencies with the visual data.

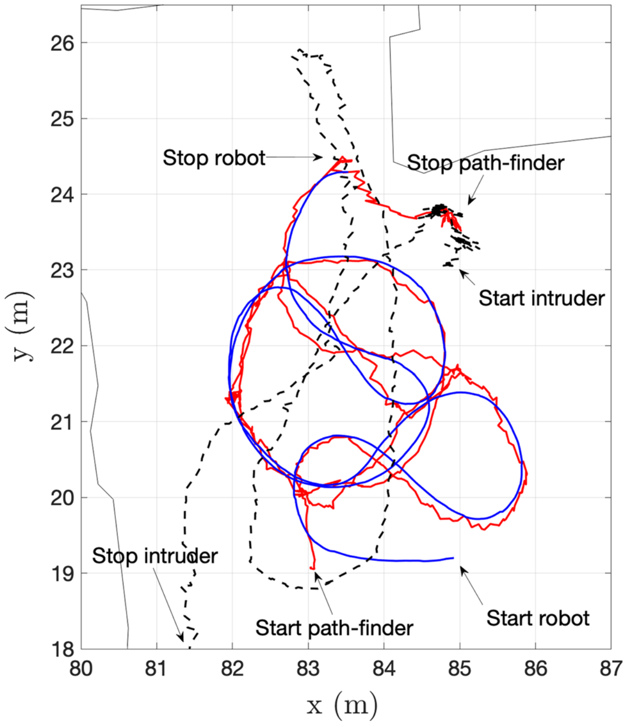


Fig. 10 Trajectory travelled by the robot in EXP- 04 (blue line) and the measured path-finder path (red line), with an additional person acting as dynamic obstacle (black dashed line) (Color figure online)

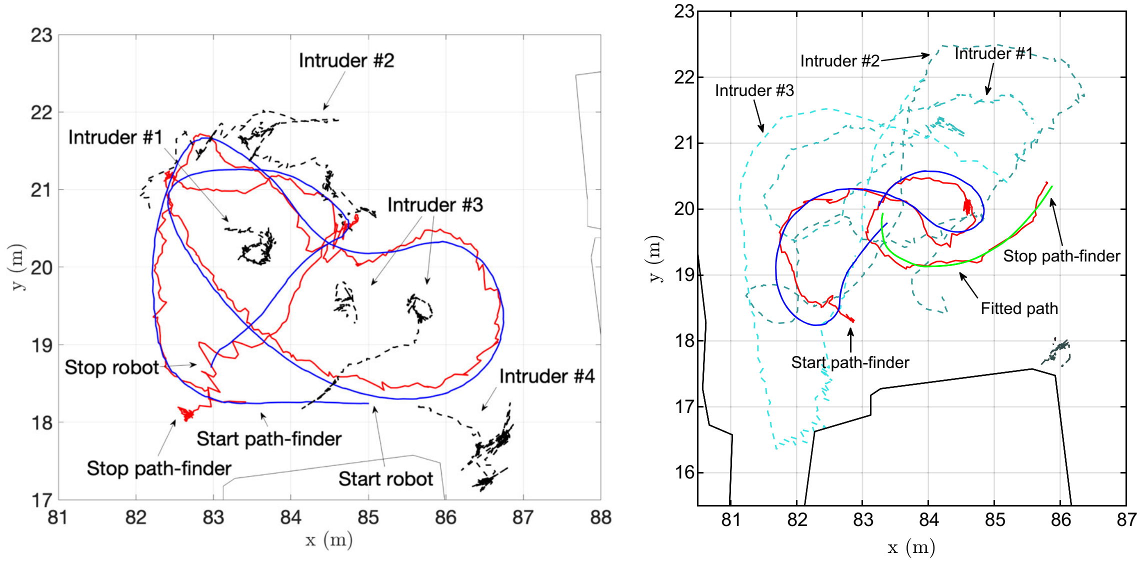
5.2.4 Crowded conditions

In a crowded scenario, the robot follows its path-finder along a winding path, meeting several other people in the same hallway. In the first case (EXP- 05), the intruders made limited movements (quasi static condition), while the robot navigates between them. As reported in Fig. 11a the resulting robot trajectory is consistent with that of the path-finder.

In a more challenging case (EXP- 06), also the intruders performed random trajectories, repeatedly obstructing the robot and occluding the path-finder but without compromising the correct robot path following (see Fig. 11-b).

5.2.5 Discerning the similarity between the path-finder and the intruder

Moreover, we tested the system safety (EXP- 07, included in the multimedia complementary material) and the tracking



(a)

(b)

Fig. 11 (a) Tracking experiments with the path-finder (red line) and other persons in a crowded environment. In EXP- 05 **a** multiple intruders (black dashed lines) are predominantly acting as bystanders, while in EXP- 06 **b** they follow random paths (coloured dashed lines). The green line depicts the fitted path which will be followed by the robot (Color figure online)

performance of our system with two people sharing highly similar visual features, and reported the results expressed in the robot local reference frame $\langle L\rangle$ (EXP- 08 of Fig. 12a).

The screenshots of the camera tracking (Fig. 12b) show moments in which the path-finder is correctly identified and the second person is classified as negative (top snapshot) and moments in which the intruder is misclassified as the path-finder (bottom). Nonetheless, even in presence of misclassification the sensor fusion with the LIDAR comes to rescue and the tracker correctly follows the path-finder (see the trajectories in Fig. 12a). Further experimental evidences of the effectiveness of the approach can be found in the video accompanying this paper.

5.2.6 Navigation without the map

Finally, we tested the behaviour of the system in an a-priori unknown environment, thus renouncing to the availability of a global localisation system. Hence, in EXP- 09, the path-finder and the robot positions are not expressed in the global frame $\langle W\rangle$ but in an arbitrary reference frame $\langle O\rangle$. Figure 13 reports the navigation task performed using only the odometry, along with the actual global localisation for comparison purposes and to overlay the odometry information on the environment map.

The obtained trajectories (blue dashed line) show that odometry localisation trivially drifts; nonetheless, the actual path travelled by the robot (depicted with the blue solid line) accurately follows the path shown by the human (solid red line). This remarkable result shows how planning in an unknown environment can be effectively executed by the robot with the path-finder. Moreover, as can be observed in the left side of Fig. 13, the navigation in the relative frame $\langle O\rangle$ turns to be more robust than in $\langle W\rangle$, which is useless when the robot is outside the a-priori known map (unless a SLAM solution is adopted).

5.2.7 Quantitative evaluation

In order to give quantitative evidence of the satisfaction of requirements **Q1** and **Q2** introduced in Sect. 1, we defined two metrics. Requirement **Q1** was measured by the percentage ratio $r_1\%$ between the camera frames where the path-finder is visible in the field of view of the camera over the total number of frames. We also report the ratio $r_2\%$ between the number of frames where the path-finder is correctly tracked over the number of frame where s/he is visible, as an additional validating metric of the reliability of the vision module. The safety constraint **Q2** was evaluated by means of the curvilinear distance s between the path-finder and the robot along the clothoid curve. From the values of $r_1\%$ reported in Table 2 we can immediately notice that, as a major difference with respect to other visual servoing works presented in the literature, in our experiments the path-finder was not always detected by the camera. For instance, in EXP- 01 and EXP- 02, which took place in a long corridor, about 30% of the camera frames had no path-finder in sight, and yet the average recognition accuracy $r_2\%$ was fairly good.

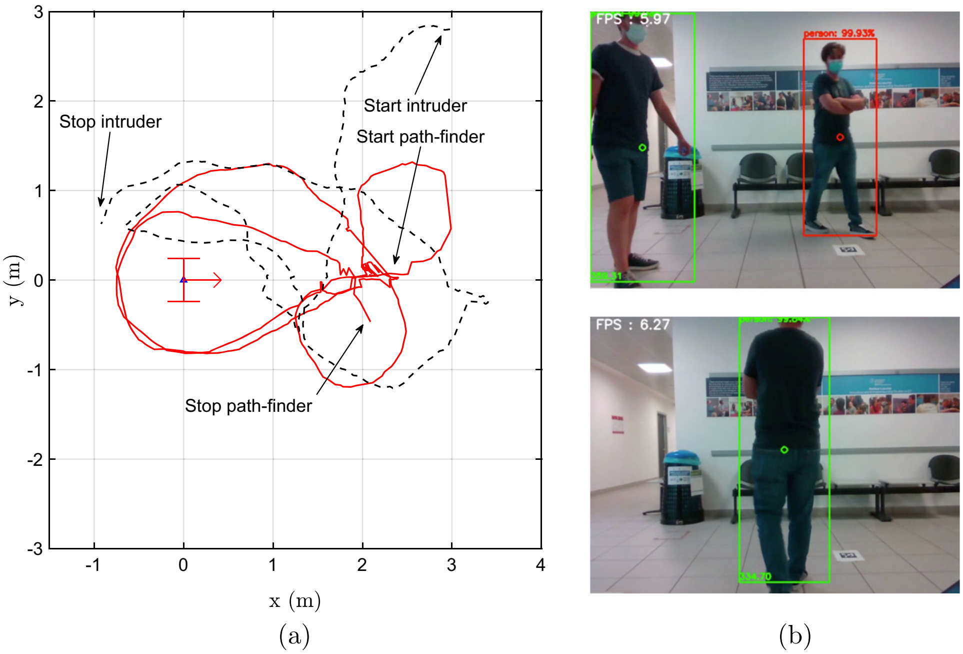


Fig. 12 **a** Tracking experiment with the path-finder (red line) and an intruder (black dashed line) with highly similar visual features for EXP- 08, expressed in $\langle L \rangle$. **b** Screenshot of a positive match of the path-finder (top) and a wrong classification of the intruder (bottom), which is recovered by the sensor fusion (Color figure online)

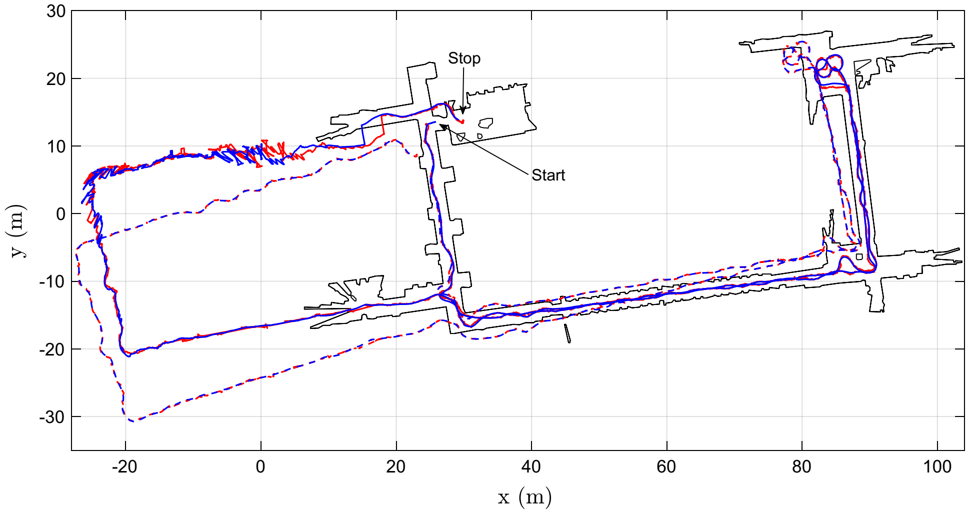


Fig. 13 Trajectory travelled by the robot in (blue lines) and the measured path-finder path (red lines), in the global localisation frame $\langle W \rangle$ (solid lines) and in the relative localisation frame (dashed line) (Color figure online)

In EXP- 04, the ratio $r_1\%$ drops to 10% due to the tortuous path taken by the path-finder. Despite this, the tracking was performed correctly (see Fig. 10), and the recognition accuracy remained very good. In EXP- 08, due to similarity between the two persons in the scene, a higher number of wrong detections occurred. For that reason, we also reported the ratio between the number of correct matches and the total number of detections. As could be expected, the ratio was lower than the one reported with the other experiments; however, we recall that with the feedback strategy of our pipeline the recognition module can overcome wrong matches of the path-finder (see Fig. 12a).

The results reported in Table 2 show that also the **Q2** requirement is satisfied in all the different scenario evaluated. In EXP- 01 and EXP- 02 the minimum distances are appropriately about 2 meters, and the path-finder is correctly followed even with distances above 5 m. The average curvilinear distance is maintained between 1 and 3 m, which ensures safety in addition to being widely recognised as the preferred spatial distance for social interactions (Rios-Martinez et al. 2015; Antonucci et al. 2019). More dangerous values can be found for the EXP- 07, since the experiment was specifically designed to trigger the robot emergency stop; nevertheless, the minimum distance reached is above 50 cm. The very high values reached by the maximum s distance in EXP- 04, EXP- 05 and EXP- 06, i.e., experiments with tortuous trajectories, are due to the obstruction of the intruder persons on the robot path.

Table 2 Summary of the two metrics for requirements **Q1** and **Q2** among the experiments reported in this paper

Exp	In-frame ratio $r\%$ (**Q1**)		Curvilinear distance s (**Q2**)		
	visible/total $r_1\%$	correct/visible $r_2\%$	avg s	min s	max s
EXP-01	67.6%	83.7%	3.970	2.632	5.150
EXP-02	78.2%	92.6%	3.328	2.015	4.486
EXP-03	–	–	1.913	1.249	2.348
EXP-04	10.9%	68.6%	3.209	1.302	4.914
EXP-05	39.8%	79.9%	2.071	0.722	3.188
EXP-06	–	–	2.466	0.860	4.271
EXP-07	–	–	1.005	0.568	1.642
EXP-08	57.5%	86.0%, (62.8%)*	–	–	–

* $r_2\% =$ correct/tracking

6 Conclusions and future work

In this paper, we have presented an approach for guiding a robot across a natural environment populated with a number of intruders. A human operator takes the role of a path-finder and the robot follows, moving in a close neighbourhood of the path physically marked by the human with her/his footsteps. This application required a combination of state-of-the-art techniques for robust perception and path reconstruction. The experimental scenarios were chosen to challenge the system's robustness and reliability (e.g., with frequent obstructions) guaranteeing at the same time the continuous tracking of all the actors in the scene and the repeatability of the experiments.

Different points remain open and are reserved for future work. A first important direction is a theoretical study of how the interaction between model based approaches and neural networks can produce results with a guaranteed accuracy for people tracking. Another interesting issue is the use of wearable haptic bracelets (Che et al. 2020) and the implementation of a protocol that the robot can use to notify to its path-finder the occurrence of exceptional conditions (e.g., when the path is too close to an obstacle and the robot cannot follow it within appropriate safety margins). Finally, we are planning to investigate how the path information can be shared among multiple vehicles for transfer learning even without any a-priori map knowledge.

Acknowledgements This work was supported by the European Union under NextGenerationEU (FAIR - Future AI Research - PE00000013).

Author Contributions Each author contributed equally to the conception, development and implementation of the idea underlying this publication. Moreover, each Author has been equally involved in the writing and revision of the manuscript.

Funding Open access funding provided by Università degli Studi di Trento within the CRUI-CARE Agreement.

Declarations

Ethical approval We certified that there is no ethical issue related to this publication.

Conflict of interest There is not conflict of interest nor competing interests regarding this publication.

References

Andreetto, M., Divan, S., Fontanelli, D., & Palopoli, L. (2017). Harnessing steering singularities in passive path following for robotic walkers. In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2426–2432. IEEE.

Antonucci, A., Bevilacqua, P., Palopoli, L., Boldrer, M., & Fontanelli, D. (2019). Motion planning in crowds: Proxemics as a base for a socially acceptable behaviour. In: *2019 I-RIM Conference*, pp. 15–16. https://doi.org/10.5281/zenodo.4782236. I-RIM.

Antonucci, A., Magnago, V., Palopoli, L., & Fontanelli, D. (2019). Performance assessment of a people tracker for social robots. In: *2019 IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*, pp. 1–6. IEEE.

Arechavaleta, G., Laumond, J.-P., Hicheur, H., & Berthoz, A. (2008). On the nonholonomic nature of human locomotion. *Autonomous Robots, 25*(1), 25–35.

Babenko, B., Yang, M.-H., & Belongie, S. (2010). Robust object tracking with online multiple instance learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 33*(8), 1619–1632.

Bertolazzi, E., Bevilacqua, P., & Frego, M. (2018). Clothoids: a C++ library with Matlab interface for the handling of clothoid curves. *Rendiconti del Seminario Matematico—Politecnico di Torino, 76*(2), 47–56.

Bertolazzi, E., & Frego, M. (2018). On the G2 Hermite interpolation problem with clothoids. *Journal of Computational and Applied Mathematics, 341*, 99–116.

Bertolazzi, E., & Frego, M. (2018). Interpolating clothoid splines with curvature continuity. *Mathematical Methods in the Applied Sciences, 41*(4), 1723–1737.

Bevilacqua, P., Frego, M., Fontanelli, D., & Palopoli, L. (2018). Reactive planning for assistive robots. *IEEE Robotics and Automation Letters, 3*(2), 1276–1283. https://doi.org/10.1109/LRA.2018.2795642

Bolme, D.S., Beveridge, J.R., Draper, B.A., & Lui, Y.M. (2010). Visual object tracking using adaptive correlation filters. In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 2544–2550. IEEE.

Cao, Z., Hidalgo, G., Simon, T., Wei, S.-E., & Sheikh, Y. (2019). Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 43*(1), 172–186.

Chen, B. X., Sahdev, R., & Tsotsos, J. K. (2017). Integrating Stereo Vision with a CNN Tracker for a Person-Following Robot. In M. Liu, H. Chen, & M. Vincze (Eds.), *Computer Vision Systems* (pp. 300–313). Springer.

Chen, B.X., Sahdev, R., & Tsotsos, J.K. (2017). Person following robot using selected online ada-boosting with stereo camera. In: *2017 14th Conference on Computer and Robot Vision (CRV)*, pp. 48–55. IEEE.

Che, Y., Okamura, A. M., & Sadigh, D. (2020). Efficient and trustworthy social navigation via explicit and implicit robot-human communication. *IEEE Transactions on Robotics, 36*(3), 692–707.

Cleveland, W. S. (1979). Robust locally weighted regression and smoothing scatterplots. *Journal of the American Statistical Association, 74*(368), 829–836.

Doisy, G., Jevtic, A., Lucet, E., & Edan, Y. (2012). Adaptive person-following algorithm based on depth images and mapping. In: *Proceedings of the IROS Workshop on Robot Motion Planning*, vol. 20.

Eisenbach, M., Vorndran, A., Sorge, S., & Gross, H.-M. (2015). User recognition for guiding and following people with a mobile robot in a clinical environment. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3600–3607.

Farina, F., Fontanelli, D., Garulli, A., Giannitrapani, A., & Prattichizzo, D. (2017). Walking ahead: The headed social force model. *PloS One, 12*(1), 0169734.

Germa, T., Lerasle, F., Ouadah, N., & Cadenat, V. (2010). Vision and RFID data fusion for tracking people in crowds by a mobile robot. *Computer Vision and Image Understanding, 114*(6), 641–651.

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 580–587.

Gupta, M., Kumar, S., Behera, L., & Subramanian, V. K. (2017). A novel vision-based tracking algorithm for a human-following mobile robot. *IEEE Transactions on Systems, Man, and Cybernetics: Systems, 47*(7), 1415–1427.

He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask r-cnn. In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2961–2969.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Identity mappings in deep residual networks. In: *European Conference on Computer Vision*, pp. 630–645. Springer.

Hearst, M. A., Dumais, S. T., Osuna, E., Platt, J., & Scholkopf, B. (1998). Support vector machines. *IEEE Intelligent Systems and their Applications, 13*(4), 18–28.

Held, D., Thrun, S., & Savarese, S. (2016). Learning to track at 100 fps with deep regression networks. In: *European Conference on Computer Vision*, pp. 749–765. Springer.

Henriques, J.F., Caseiro, R., Martins, P., & Batista, J. (2012). Exploiting the circulant structure of tracking-by-detection with kernels. In: *European Conference on Computer Vision*, pp. 702–715. Springer.

Hess, W., Kohler, D., Rapp, H., & Andor, D. (2016). Real-time loop closure in 2D LIDAR SLAM. In: *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1271–1278.

Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., & Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861.

Islam, M. J., Hong, J., & Sattar, J. (2019). Person-following by autonomous robots: A categorical overview. *The International Journal of Robotics Research, 38*(14), 1581–1618.

ISO13482 (2014). Robotic devices–safety requirements for personal care robots. International Organization for Standardization.

Jiang, S., Zhang, J., Zhang, Y., Qiu, F., Wang, D., & Liu, X. (2018). Long-term tracking algorithm with the combination of multi-feature fusion and yolo. In: *Chinese Conference on Image and Graphics Technologies*, pp. 390–402. Springer.

Kalal, Z., Mikolajczyk, K., & Matas, J. (2010). Forward-backward error: Automatic detection of tracking failures. In: *2010 20th International Conference on Pattern Recognition*, pp. 2756–2759. IEEE.

Kalal, Z., Mikolajczyk, K., & Matas, J. (2011). Tracking-learning-detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 34*(7), 1409–1422.

Koide, K., & Miura, J. (2016). Identification of a specific person using color, height, and gait features for a person following robot. *Robotics and Autonomous Systems, 84*, 76–87.

Lam, C.-P., Chou, C.-T., Chiang, K.-H., & Fu, L.-C. (2010). Human-centered robot navigation-towards a harmoniously human-robot coexisting environment. *IEEE Transactions on Robotics, 27*(1), 99–112.

Layne, R., Hospedales, T.M., Gong, S., & Mary, Q. (2012). Person re-identification by attributes. In: *Bmvc*, vol. 2, p. 8.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A.C. (2016). Ssd: Single shot multibox detector. In: *European Conference on Computer Vision*, pp. 21–37. Springer.

Lukezic, A., Vojir, T., Čehovin Zajc, L., Matas, J., & Kristan, M. (2017). Discriminative correlation filter with channel and spatial reliability. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6309–6318.

Markis, A., Papa, M., Kaselautzke, D., Rathmair, M., Sattinger, V., & Brandstötter, M. (2019). Safety of mobile robot systems in industrial applications. In: *Proceedings of the ARW & OAGM Workshop*, pp. 26–31.

Munkres, J. (1957). Algorithms for the assignment and transportation problems. *Journal of the Society for Industrial and Applied Mathematics, 5*(1), 32–38.

Neubeck, A., & Van Gool, L. (2006). Efficient non-maximum suppression. In: *18th International Conference on Pattern Recognition (ICPR'06)*, vol. 3, pp. 850–855. IEEE.

Nguyen, T.-M., Cao, M., Yuan, S., Lyu, Y., Nguyen, T.H., & Xie, L. (2021). Viral-fusion: A visual-inertial-ranging-lidar sensor fusion approach. IEEE Transactions on Robotics.

Nikdel, P., Shrestha, R., & Vaughan, R. (2018). The hands-free push-cart: Autonomous following in front by predicting user trajectory around obstacles. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4548–4554. IEEE.

Pun, C.-M., Yuan, X.-C., & Bi, X.-L. (2015). Image forgery detection using adaptive oversegmentation and feature point matching. *IEEE Transactions on Information Forensics and Security, 10*(8), 1705–1716.

Quispe, R., & Pedrini, H. (2019). Improved person re-identification based on saliency and semantic parsing with deep neural network models. *Image and Vision Computing, 92*, 103809.

Redmon, J., & Farhadi, A. (2018). Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767.

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 779–788.

Rios-Martinez, J., Spalanzani, A., & Laugier, C. (2015). From proxemics theory to socially-aware navigation: A survey. *International Journal of Social Robotics, 7*(2), 137–153.

Ristani, E., & Tomasi, C. (2018). Features for multi-target multi-camera tracking and re-identification. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6036–6046.

Roth, P. M., Hirzer, M., Köstinger, M., Beleznai, C., & Bischof, H. (2014). Mahalanobis distance learning for person re-identification. In S. Gong, M. Cristani, S. Yan, & C. C. Loy (Eds.), *Person Re-Identification. Advances in Computer Vision and Pattern Recognition* (pp. 247–267). Springer.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2015). Going deeper with convolutions. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–9.

Wang, M., Liu, Y., Su, D., Liao, Y., Shi, L., Xu, J., & Miro, J. V. (2018). Accurate and real-time 3-d tracking for the following robots by fusing vision and ultrasonar information. *IEEE/ASME Transactions on Mechatronics, 23*(3), 997–1006.

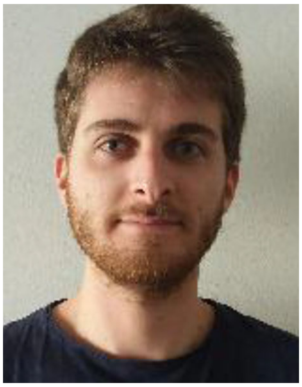
Wolcott, R.W., & Eustice, R.M. (2014). Visual localization within lidar maps for automated urban driving. In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 176–183. IEEE.

Zheng, L., Shen, L., Tian, L., Wang, S., Wang, J., & Tian, Q. (2015). Scalable person re-identification: A benchmark. In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1116–1124.

Zhen, W., Hu, Y., Liu, J., & Scherer, S. (2019). A joint optimization approach of lidar-camera fusion for accurate dense 3-d reconstructions. *IEEE Robotics and Automation Letters, 4*(4), 3585–3592.

Zhou, H., Taj, M., & Cavallaro, A. (2008). Target detection and tracking with heterogeneous sensors. *IEEE Journal of Selected Topics in Signal Processing, 2*(4), 503–513.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Alessandro Antonucci received the M.S. in Mechatronic Engineering and the Ph.D. degree in information and communication technology from University of Trento, Trento, Italy, in 2018 and 2022, respectively. He is currently a R&D developer at Polytec Intralogistics SRL. His research interests focus on human-robot interaction and pedestrian motion tracking and prediction.

Paolo Bevilacqua received the master's degree in computer science and the Ph.D. degree in information and communication technology from University of Trento, Trento, Italy, in 2015 and 2019, respectively. He is currently a cofounder of Polytec Intralogistics SRL where he develops solutions for shared warehouse automation. His current interests focus mainly on motion planning for wheeled robots, and on socially compliant autonomous navigation in environments shared with humans.

Stefano Leonardi received the master's degree in computer science and the Ph.D. degree in information and communication technology from University of Trento, Trento, Italy, in 2018 and 2020, respectively. He is currently as a junior software and learning engineering at TVEyes Language Technology Srl.

Luigi Paolopoli is Full Professor of Computer Engineering at the "Dipartimento di Ingegneria e Scienza dell'Informazione (DISI)", University of Trento. He received a PhD degree from "Scuola Superiore S. Anna, Pisa" in 2002. His research interests include real-time embedded control, formal methods and stochastic analysis of real-time systems.

Daniele Fontanelli received the M.S. degree in Information Engineering in 2001, and the Ph.D. degree in Automation, Robotics and Bioengineering in 2006, both from the University of Pisa, Pisa, Italy. He was a Visiting Scientist with the Vision Lab of the University of California at Los Angeles, Los Angeles, US, from 2006 to 2007. From 2007 to 2008, he has been an Associate Researcher with the Interdepartmental Research Center "E. Piaggio", University of Pisa. From 2008 to 2013 he joined as an Associate Researcher the Department of Information Engineering and Computer Science and from 2014 the Department of Industrial Engineering, both at the University of Trento, Trento, Italy, where he is now a Full Professor. He has authored and co-authored more than 180 scientific papers in peer-reviewed top journals and conference proceedings. He is currently an Associate Editor in Chief for the IEEE Transactions on Instrumentation and Measurement, an Associate Editor for the IEEE Robotics and Automation Letters and for the IET Science, Measurement & Technology Journal. From 2018 he is also an Associate Technical Program Committee Member for the IEEE/RSJ International Conference on Intelligent Robots and Systems and Senior Member of IEEE. From 2022 he is a member of IMEKO Technical Committee 17 on "Measurement for Robotics". His research interests include localisation algorithms, wheeled mobile robot control and service robotics, real-time estimation and control.