Research Article

INTERNATIONAL JOURNAL OF ADVANCED ROBOTIC SYSTEMS

International Journal of Advanced Robotic Systems
May-June 2017: 1–15

DOI: 10.1177/1729881417705923
journals.sagepub.com/home/arx
SAGE

Vision-based people detection using depth information for social robots: An experimental evaluation

Arnaud Ramey, Álvaro Castro-González, Maria Malfaz, Fernando Alonso-Martin and Miguel A Salichs

Abstract

Robots are starting to be applied in areas which involve sharing space with humans. In particular, social robots and people will coexist closely because the former are intended to interact with the latter. In this context, it is crucial that robots are aware of the presence of people around them. Traditionally, people detection has been performed using a flow of two-dimensional images. However, in nature, animals' sight perceives their surroundings using color and depth information. In this work, we present new people detectors that make use of the data provided by depth sensors and red-green-blue images to deal with the characteristics of human–robot interaction scenarios. These people detectors are based on previous works using two-dimensional images and existing people detectors from different areas. The disparity of the input and output data used by these types of algorithms usually complicates their integration into robot control architectures. We propose a common interface that can be used by any people detector, resulting in numerous advantages. Several people detectors using depth information and the common interface have been implemented and evaluated. The results show a great diversity among the different algorithms. Each one has a particular domain of use, which is reflected in the results. A clever combination of several algorithms appears as a promising solution to achieve a flexible, reliable people detector.

Keywords

Social robotics, people detection, user detection, kinect, depth sensor, depth image processing, ROS, benchmarking

Date received: 28 September 2016; accepted: 24 March 2017

Topic: Special Issue - Biologically-Inspired Vision Systems in Robotics
Topic Editor: Antonio Fernández-Caballero

Introduction

Nowadays more and more robotic applications are moving from isolated environments and laboratories to areas where robots and people coexist. Under this condition, robots need to be aware of the presence of humans around them. In social robotics (SRs), this condition is even more important since these robots are intended to interact with people that live together with them. Consequently, *people detection* is a fundamental feature in SRs.

People detection refers to the capacity of a robot to perceive the presence of a person. In this work, we aim at giving this skill to social robots, which consists of detecting the users (We apply the term *user* to refer to those people who can be engaged in human–robot interaction (HRI). Two relevant concepts are usually mixed up: *people detector* and *user detector*. Both refer to algorithms that will do

Universidad Carlos III de Madrid, Av. de la Universidad 30, Leganés, Madrid, Spain

Corresponding author:
Álvaro Castro-González, Robotics Lab, Universidad Carlos III de Madrid, Av. de la Universidad 30, Leganés, Madrid 28911, Spain.
Email: acgonzal@ing.uc3m.es

the same thing, which is finding human shapes in the sensors input. However, a user detector is more specifically used when a positive interaction between the system and the human person is possible. This can be the case in robotics, in multimedia kiosks, and so on. In this article, we both terms use indistinctly.) close to the robot, thanks to its sensors. Here we focus on people detection considering how HRI usually happens, that is, mostly within short distances and in daily, unstructured environments.

Detecting human shapes in an image stream is a problem that is related to many more fields than just SRs: for example, home security[1,2] playing video games,[3] or monitoring the activity of sick people.[4] Even in robotics, the applications are multiple: they include gesture recognition,[5] follow the leader behaviors,[6,7] turning the robot into a gaming companion,[8] and many more.

The important amount of research that has been done results in a wide range of methods and algorithms that have been proven to give satisfactory results for people detection using visual data. Traditionally, these people detection methods used two-dimensional (2-D) color information (red-green-blue (RGB) images). However, if we look at nature, animals use both color and depth information to perceive their environment. In this sense, with the uptake of depth sensors in robotics (mainly the *kinect* device), new algorithms have been developed using depth information. In this article, we present new algorithms using depth images (i.e. 3-D data) inspired on the most frequent algorithms using conventional 2-D images. Besides, algorithms from other fields have been transformed to deal with the characteristics of the scenarios where humans and robots interact.

Researchers have shown that the gap between subjects engaged in interpersonal relations is a key aspect to be socially accepted and effective when interacting,[9] so is in human–robot relations.[10] Thus, it is important that social robots are able to both detect potential users and estimate the distance to them. One important advantage of using 3-D data to detect humans is that, once a person is identified, obtaining the distance to that person is straightforward. The 3-D-based people detectors presented in this work will provide, not just the presence or absence of a person, but the location in relation to the robot.

Due to their low cost and performance, depth sensors have become a mainstream device available in most of the social robots operating in the world. These devices will provide the data needed by the people detectors implemented in this work. Considering that social robots will operate in different, unstructured, and daily environments, it is crucial that they are equipped with the required devices for their normal operation. Thus, the people detection algorithms presented here can be run in any robot equipped with depth (RGB-D) sensors.

The diversity of different people detection algorithms has lead to create ad hoc solutions to integrate them into robotic control architectures. Moreover, the different nature of the data provided by each of these algorithms makes it difficult to compare their performance. Here we propose a common interface that can be applied to any people detection method.

The rest of the article is structured as follows. First, in the section on related works, the most relevant algorithms for user detection existing in the different above-mentioned fields will be reviewed. Second, using the ROS framework, we present a common interface for people detectors and justify its benefits (section on defining a common interface for people detectors). All the methods developed in this work make use of this interface. Then, our contributions to user detection will be detailed in the section on people detectors using depth information. These contributions include, on the one hand, the development of new algorithms using 3-D data built on existing conventional, color image–based algorithms; on the other hand, several people detection algorithms developed to be applied in other fields (such as gaming, or pedestrian detection) have been adapted to HRI scenarios. After, the performance of the different implemented methods is compared in the section on benchmarking people detectors. In the discussion section, we explain the results, the strengths, and the weaknesses of each method. Finally, we summarize the work presented in this article (conclusion section).

Related works

As said before, the detection of human shapes in a stream of images is a problem that has been tackled by researchers several times in the past. Many articles propose ad hoc solutions, that is, the user detection requires a nonnatural modification of the environment or of the interaction flow (e.g. a specific color of clothing[11] or visual markers[12]).

Considering that social robots operate in daily environments, we discard this category and focus on generic (non-ad hoc) detection algorithms using visual information (2-D and 3-D). That is, algorithms that do not imply a modification of the environment or the people to be detected and that use the information provided by a camera (e.g. RGB or RGB-D). Among the many existing algorithms, a limited number of them will be reviewed. These are the ones that are, at the same time, the most accurate and the most relevant to the needs of SRs.

Lately, convolutional neural networks (CNN), or deep learning, have become a very popular technique for many computer vision problems, such as people detection.[13] They present a very good accuracy, but, however, it is known that CNN are very slow at inference time.[14–16] Unfortunately, the real-time constraint is mandatory in HRI scenarios. and robots have limited onboard resources. Furthermore, deep learning algorithms require a substantial amount of training data, while data sets offering both color and depth data and with such an amount of people images involved in HRI scenarios do not exist. For these reasons, these methods are not considered in this work.

Face detection

Face detection consists of determining if human faces are visible in a video stream and, if it is so, what are their position in the image. This is one of the classical challenges in vision.

Nowadays, some fairly standard techniques are available for face detection and give a very good accuracy rate. The technique presented by Viola–Jones[17] is one of the most used in robotics and its lightweight computational cost allows a high-frequency analysis of the video stream. The Viola–Jones object detection framework can actually be trained to detect any object,[18] but it is especially popular for face detection.

The method of Viola and Jones is an example of supervised learning. The learning is based on the appearance of the training images. The process consists of seizing the content of each image by computing the so-called *characteristics* in rectangular zones of the image that overlap each other. These characteristics are a synthetic and descriptive representation of the values of the pixels and are more efficient to be dealt with.

The second key element in Viola and Jones is the use of a boosting method in order to select the best characteristics. *Boosting* is a technique that enables the building of a *strong* classifier with a linear combination of *weak* classifiers. In this method, characteristics are seen as weak classifiers. Hence, the learning process of the weak classifier only consists of learning the threshold value of the characteristic so as to split better positive samples from negatives. The original detector uses three different characteristics, while the modified Lienhart and Maydt detector[19] adds two others, and includes two diagonal orientations.

Another very widespread face detection algorithm is the detector based on neural networks[20]. It only works well with frontal, upright faces. Thanks to its accuracy, after its release in 1998, this algorithm became the most widespread and is used in numerous works of the following years (see for instance[21]) till the publication of the previously presented Viola-–Jones detector. The latter being more accurate and about 15 times faster,[17] therefore, becoming the weapon of choice.

Human body detectors: Histogram of oriented gradients

A histogram of oriented gradients (HOG) is a feature used in computer vision for object detection. It has turned out to be a very efficient technique for the detection of human shapes. It was presented first by Dalal and Triggs.[22] It is a 2-D algorithm using an RGB image as input and returns as output the rectangular estimations of the people, and so it does not need, neither uses, a depth image. It makes use of histograms of *image gradients*.

The HOG algorithm is based on the computation of local histograms of the gradient orientation called *HOG descriptors*. The concept of the HOG descriptor is that the distribution of the gradient intensity or the direction of the edges can describe the appearance and the shape of a physical object in an image.

To compute the HOG descriptors of an image, first, the image is divided into a continuous grid of small areas called *cells*. In each cell, for each pixel of the cell, the directions of the gradients is computed. A histogram of all these directions is then built. The HOG descriptor is then made of a set of these local histograms (one histogram per cell).

The HOG algorithm then needs data for supervised learning. For this purpose, a data set of images is created. For each image, the presence of a person in it is manually labeled. The HOG descriptors are computed on each image, and a simple binary SVM classifier[23] is trained with them for classification. Note that the SVM classifier is a nonsophisticated classifying algorithm on purpose, so as to demonstrate the efficient description of the objects by the HOG descriptors.

The original HOG detector has been reused and improved by several authors. Two speedups in the computation, but without considerable accuracy improvement, can be mentioned: first, researchers of Mitsubishi have coupled the same descriptors with cascade classifiers, similar to the ones used by Viola and seen previously in the subsection on face detection, leading to an up to 70 times speedup, but their improvement is protected by a patent[24]; and second, subcell interpolation computes efficiently the HOG descriptors per block, and the reuse of the features between overlapping blocks leads to a further 5 times speedup.[25]

Kinect API: NiTE

The patented PrimeSense NiTE middleware allows detecting and tracking human shapes from depth maps. It is the piece of software that powers the detection and tracking of the users for the Microsoft Kinect device developed for the XBox 360.[26]

Although the technique employed and the source code are not available, it is likely that motion analysis and clustering techniques are at the core. Indeed, detection of a human player is activated by her motion. In other words, a still user is not detected, and a moving object will be detected as a human user if it has similar dimensions.

The NiTE middleware supplies three data for higher-level applications: the RGB image, the depth image, and the multimask. The *multimask* is a one-channel byte image. This image, synchronized with the RGB and the depth ones, indicates where the users are: if a pixel p of the users mutimask has a value of 0, it means there is no user in p, while if it has a value of 1, p corresponds to a pixel of the user 1, and so on. For a given user, a *user mask* is the multimask image where all pixels that do not belong to this user are set to 0 (i.e. "erasing" the other users).

Figure 1. Several social robots equipped with a depth sensor.

Polar-Perspective Map

Another approach for people detection, thanks to depth images presented by Howard and Matthie,[27] where it is part of a complete pedestrian detection system only based on stereovision.

The authors use a 2-D occupancy map with a polar resolution called polar-perspective map and abbreviated as PPM. The so-called PPM is an occupancy map based on the polar coordinate system: It uses a regular grid based on the bearing of the points and their inverse distance to the device. As such, the resolution of closer points is higher than of far points: Unlike the Cartesian map, it can be tuned to match exactly the resolution of the device.

The 3-D point cloud of the environment is projected into the PPM: for every point of the cloud, the corresponding cell in the map increases by one. The clusters of the map with a high accumulation factor are produced by vertical objects in the original point cloud. They correspond to regions of interest: they are segmented from the original image. Then a simple Bayes classifier determines if the 2-D shape of each cluster is similar to a human shape. The system is compatible with classical image processing techniques, such as appearance-based algorithms.

Stereovision is, according to authors, a very liable solution for navigation and pedestrian detection. However, their system is designed for stereo cameras mounted on outdoors vehicles, with ranges between 5 and 50 m. This makes difficult to use it for indoor robotics platforms.

Defining a common interface for people detectors

Usually it is desirable to be able to run the same software modules in different robotic platforms. In our case, we would like to be able to use several people detectors in our social robots (Figure 1).

To ease this process, a common practice in computer science consists of standardizing the communication layer and the data that are exchanged by the different modules, better than the proper structure of these modules. For this reason, we designed a common data structure that contains the data of a detected user and will be filled by any user detector that we want to use in our robots.

We called this data structure PeoplePose (PP). It corresponds to a single user detection and contains all the information worth being shared, such as the 3-D position and orientation of the user or the confidence of the detection.

Considering that the robot operating system (ROS) framework has become the de facto standard in robotics, the PP data structure is implemented as a ROS msg.[28] The msg formalism is in fact a simple message description language for describing data structure exchanged by ROS nodes. The details of the PP message are in code listing 1.

Code listing 1: "The PP message. Note the ':' characters describe the inner fields of a field"

```
// the header, useful for the stamp and the frame
std_msgs/Header header
:time stamp
:string frame_id

// the estimated position and orientation of this person
geometry_msgs/Pose pose
:geometry_msgs/Point position
::float64 x, y, z
:geometry_msgs/Quaternion orientation
::float64 x, y, z, w

// the confidence (between 0=really unsure and 1=very sure)
float32 confidence

// the standard deviation of the estimated pose
float32 std_dev

// the color mask of the user: a tight crop of the RGB image to the user
      ↪ detection.
sensor_msgs/Image rgb

// the depth mask of the user
sensor_msgs/Image depth

// the binary mask of the user.  The image pixels are = 0 where the user is not,
// and >0 where she is (that is, any point of her body).
sensor_msgs/Image user

// a list of attributes of this person, for instance her height, her preferences
      ↪ ...
string[] attributes_names

// the values of the attributes definited in attributes_names
string[] attributes_values

// the supposed name of recognized people (for instance, "Bob"). Only filled by
      ↪ user recognition methods, such as face recognition or multimodal
      ↪ fusion.
string person_name
```

The PP data type starts with a header that provides a time stamp. Then, the 3-D position and orientation of the user is considered by the pose field whose type is geometry_msgs/Pose. This type of data is commonly used as the output of many detectors but, here, we extend it with additional information relevant in HRI. The confidence field represents the certainty of the information provided by the detector. In case of using multiple detectors, the standard deviation resulting from the fusion process fills the std_dev field, and three user images are considered in this message: the RGB, the depth, and the binary masks of the user.

The PP message considers the possibility of containing several user's attributes (see attributes_name and attributes_values fields), such as her height, age, or mood. These attributes are not mandatory and should be provided by the people detector.

Moreover, some fields imply higher-level understanding of the scene and, consequently, some fields will be filled by advanced user detection algorithms. For instance, the person_name field already implies some user recognition routine that most user detection algorithms do not have. For

this reason, in this work, this field is set at the default value NO_RECOGNITION_MADE most of the time.

The PP message is intended to be usable by any kind of user detector. However, all fields of the message will not be filled by all methods: for instance, a user detector based on the information of a 2-D range finder will not use the image fields.

With one given data input, a detector can detect several users at once. For instance, a face detector can find several users in the same RGB picture. In order to allow such feature, the different PP generated by each detection are then gathered into a single message: this collection of PPs is then called a PeoplePoseList (PPL). In addition to the list of PP, the PPL message also contains the name of the algorithm used by the detector and a time stamp. The full structure is visible in code listing 2.

```
std_msgs/Header header
// the name of the method used, ex: "face_recognition_eigen"
string method
// the list of found poses
people_msgs/PeoplePose[] poses
```

Code listing 2: The PPL message

An algorithm capable of generating PPL messages is called a PeoplePoseList Publisher (PPLP). All the people detectors running in the robotic architecture that controls our social robots will then exchange this common data structure PPL.

People detectors using depth information

Usually, HRI happens when robots and people are close enough to establish a dialog or when a person is in the surroundings of the robot. In these situations, robots do not use to get around but stand while communicating with people. Besides, since social robots are intended to be deployed in everyday environments, it is desirable to avoid external sensors and endow the robots with all the required devices for a normal operation.

In this section, we present five people detection algorithms that take into account the above circumstances and use depth and color information to locate users around the robot. They are integrated in our robot control architecture as PPLPs, which are detailed in the following subsections. The next PPLPs are inspired in previous works described in the section on related works.

Our interest in using 3-D data for people detection lies in the fact that the computation of the user location is straight once she has been detected. This simplifies the software, making it lighter and easier to deploy.

Face detection-based PPLP with depth information

The classical 2-D Viola–Jones and neural network detectors presented in the subsection on face detection only use the color (RGB) image data and indicate the position of the found faces in the image frame, by their bounding boxes, in pixels. Here, we propose a new algorithm that, based on the geometrical shape of a human head, detects faces that comply with certain dimensions. The underlying idea is the following: the 3-D points corresponding to a face complies with certain geometric constraints, especially in the width and height of their 3-D bounding box. Indeed, two points belonging to one given face cannot be away from one of the other of, say, more than 1 m.

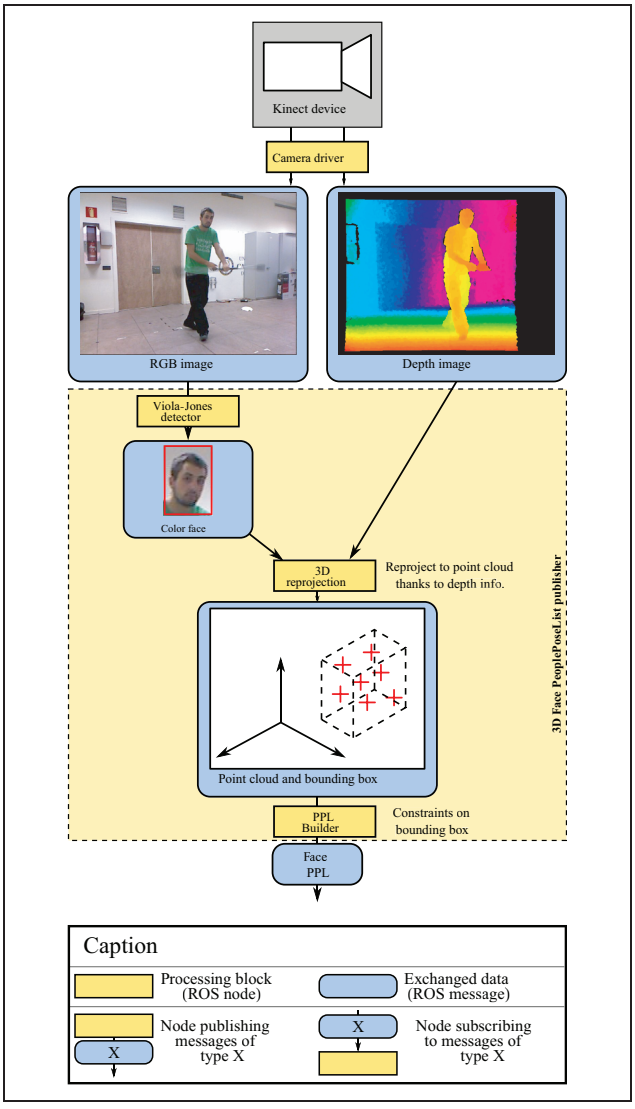

Figure 2. Processing pipeline for the three-dimensional (3-D) face detection algorithm.

As such, to determine if a zone of the image classified as a face by the Viola–Jones detector is really a face, we will sample a given number of 2-D points from this zone, that we will reproject to 3-D using the depth (distance) image. If the bounding box of these reprojected points does not comply with generic given geometric constraints, this detection is classified as a false positive and discarded. The pipeline is illustrated in Figure 2.

The face detection PPLP was implemented in C++, using the Viola–Jones implementation supplied by

OpenCV.[29] The RGB and depth data are supplied by the kinect drivers. The thresholds for the dimensions of an acceptable face bounding box were determined experimentally: the maximum bounding-box depth of the face points is set to 30 cm and its height to 40 cm. Note these thresholds are reconfigurable dynamically, thanks to the mechanism of ROS parameters.

For each face detection, an output PP message is built: the RGB, depth, and user masks are obtained via a propagation from seed. In this method, the seed is the 2-D center of the face detection. The 3-D reprojection of this same seed is used as the output user position.

The propagation from seed generates a full user mask given two inputs: the depth image and a so-called *seed* pixel, which is in fact a pixel position that we know for sure that it belongs to a user.

The idea is to run a propagation algorithm in the depth image, starting from the seed, and stopping at the depth edges, which are pixels where there is an edge discontinuity. This discontinuity can be found at the edge of a user: the background floor is several meters behind the user, which generates a gap in the depth value.

The edges can be obtained in the depth image applying a Canny edge detection algorithm[30] on it: The filter is then called a Depth Canny. A flood fill propagation from the seed in the Depth Canny edge image will then include all points of the user.

The HOG-based PPLP with depth information

The2-D HOG detector (subsection on human body detectors: histogram of oriented gradients) is made for finding people in an RGB image and supplies a rough rectangular estimate of their position in that image. In our case, we are interested in their 3-D position, and we make use of the depth image.

The people detector presented in this section initially uses the 2-D HOG detector, in particular the OpenCV HOG C++ implementation.[29] The search is performed with a scaling factor of 1.05 (increases the detection window size by 1.05 for each search scale). These human locations are in fact rectangles that give a rough bounding box of the user. This information is 2-D: The rectangles indicate where the user is in pixel coordinates.

Based on this information, we have to pick a seed pixel that will determine the user position and its location after running a propagation algorithm, just as the one mentioned in the subsection on face detection-based PPLP with depth information. Using the geometric center of the 2-D rectangles is not a reliable method. The rectangles returned by the 2-D HOG detector are only rough estimations of the users: It is possible that their center pixel do not belong to the user's mask, which is a prerequisite for the success of this method.

Another approach is to consider that the user represents most of the content of the rectangle. As such, for a given detection r of the HOG detector, if r is a positive detection, that is, there is a user in r, then this user corresponds to the main 3-D cluster inside of r. In other words, the user corresponds to the main 3-D cluster in the point cloud generated by reprojecting each point of r. The processing pipeline is made of several stages. The details are given algorithm 1.

Data: RGB image rgb, depth image $depth$
Result: PPL message ppl
RectangleVector $R \leftarrow HOG\_detect(rgb)$;
initialize PointCloudVector U ;
for *Rectangle* $r \in R$ **do**
  `// reproject only ROI of the depth image`
  PointCloud $c_r \leftarrow reproject3D(depth(r))$;
  PointCloudVector $C \leftarrow euclidean\_cluster(c_r)$;
  PointCloud $u_r \leftarrow$ biggest cluster (in number of points) of C ;
  if $bounding\_box(u_r)$ *has human dimensions* **then**
    add u_r to U ;
for *PointCloud* $u \in U$ **do** `// build PPL`
  initialize `PP` pp ;
  $pp.pose \leftarrow centroid3D(u)$;
  $mask \leftarrow reproject2D(u)$;
  $pp.user \leftarrow mask$;
  $pp.rgb \leftarrow rgb(mask)$;
  $pp.depth \leftarrow depth(mask)$;
  add pp to ppl ;
return ppl ;

Algorithm 1: The processing pipeline for the HOG PPL generator.

The key step of the algorithm is the clustering of the 3-D point cloud c_r generated by reprojecting in 3-D the pixels in a HOG detection r. This is done, thanks to the point cloud library (PCL) (http://www.pointclouds.org/. The clustering makes use of a compact and fast representation of the cloud, thanks to a Kd-tree, and fast access to the neighbor points of a query, thanks to an octree. Very briefly, the algorithm goes as follows: for the first point of the cloud, creating a queue with its Euclidean neighbors. Then for each point of the queue that was not seen already, recursively add the neighbors of this point to the queue.). We used an Euclidean neighbor distance threshold of 10 cm.

The biggest cluster of the 3-D point cloud c_r should then correspond to the user shape. Constraints on the dimensions of the 3-D bounding box of this biggest cluster filter out from others the cluster corresponding to a person. These constraints are actually permissive ranges on the values in meters of the height, width, and depth of this 3-D bounding box. If one of its values is outside the thresholds for human dimensions, then the detection r is discarded. Otherwise, a PP message is built. The final 3-D position of the user corresponds to the 3-D centroid of the main cluster.

NiTE-base2-D PPLP

In this section, we present a PPLP that uses a people detector natively processing 3-D information: the NiTE

middleware for the Microsoft Kinect device seen in the subsection on kinetic API: NiTE. This section describes how it has been integrated in the robotic control architecture as a PPLP. This people detector is well-known in robotics, and it has been included in this work for comparative purposes.

The NiTE middleware supplies a data structure that is straightforward in converting into a PPL: the *users multimask*. We have seen that the unique values of the users multimask, which are strictly positive integers, can be seen as the users' IDs. The NiTE middleware already includes tracking capabilities: the same physical user, whether she is visible in successive frames, is identified by the same ID in the resulting successive multimasks.

The number of users visible by the camera is equal to the number of unique nonzero values (IDs) in the users multimask. For each ID of the users multimask, we define the *user mask* as the set of pixels in the users multimask that have a value equal to this ID. This is equivalent to setting all other IDs in the multimask to 0. Then, the 3-D position of the user with this ID is the Center of Mass (COM) of the reprojection in 3-D of this user blob. The user RGB image corresponds to the part of RGB image where the user mask is nonnull. Similarly, the depth image of the user is the intersection of the depth image and the user mask.

These properties are used for the conversion between the users multimask and the PPL message. First, we determine all the unique values (IDs) of the multimask, then for each ID, we compute the COM of its user mask. This COM is reprojected to 3-D and stored in the PPL message.

The NiTE-based PPLP is structured as a C++ ROS *node* that subscribes to the three image streams published by the NiTE middleware node: RGB, depth, and multimask. The full processing pipeline is visible in Figure 3. A sample of user detection and PPL building, thanks to the NiTE middleware, is visible in Figure 4.

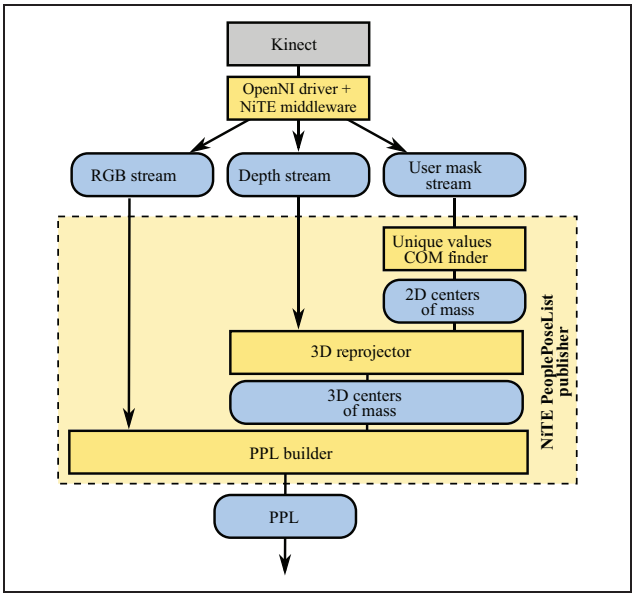

Figure 3. The NiTE-based user detector pipeline. The caption is identical to Figure 2.

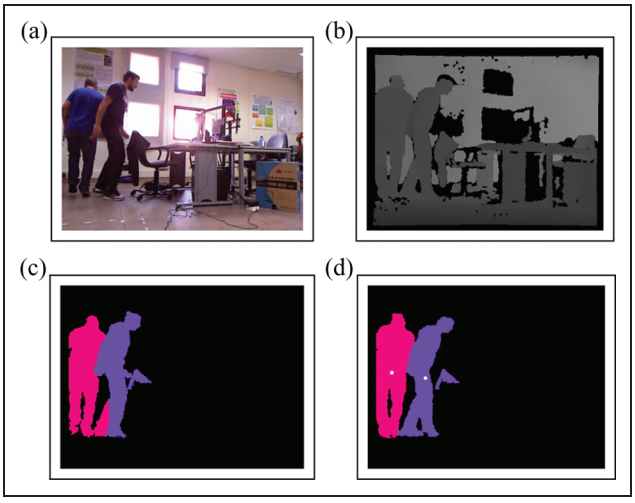

Figure 4. Sample images of the NiTE-based PeoplePoseList Publisher. (a), (b), and (c) The data supplied by the NiTE middleware, respectively, the red-green-blue (RGB) image, the depth image remapped to gray scale and the users multimask. (d) the center of mass of each user (white circle).

The PPM-based PPLP

This people detector uses the fact that a person appears as a 3-D cluster, that is, a set of points tightly close to one to another, in the 3-D point cloud given by the range-imaging device. When projecting this 3-D cloud on the ground plane, these clusters will be projected onto the same area, thus generating a sort of high-density blob on the ground plane. Standing persons can then easily be characterized by the size of the blob: their height will most likely belong to a span of characteristic human sizes, while their width corresponds to the footprint of a human user.

In the case of the original PPM, the person detection was conducted running a Bayesian classifier on the high-density blobs of the ground plane. Here, those blobs are projected back to the 3-D coordinate space and filtered out according to the size of a standing human.

We implemented the PPM-based people detector using the geometric transform from the Cartesian system of coordinates to the PPM indicated by[27] and having the following definition:

$$f : \begin{cases} \mathbb{R}^2 & \mapsto & \mathbb{R}^2 \\ \#4 & \rightarrow & (r, \theta) = \left(\sqrt{x^2 + y^2},\ \tan^{-1} \dfrac{y}{x} \right) \end{cases}$$

Besides, two resolutions are chosen to determine the size of the cells: a spatial one (corresponding to steps in r) and an angular one (steps in θ).

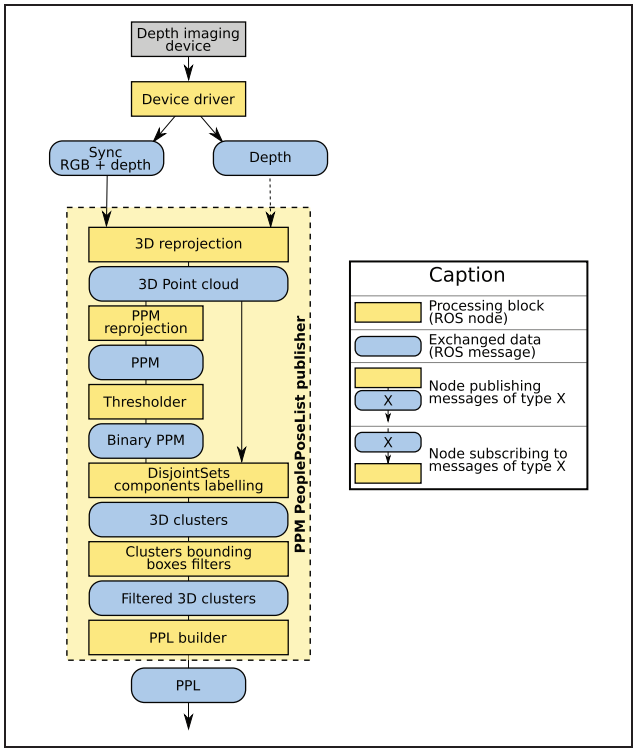

Figure 5. The polar-perspective map (PPM)-based user detector pipeline.

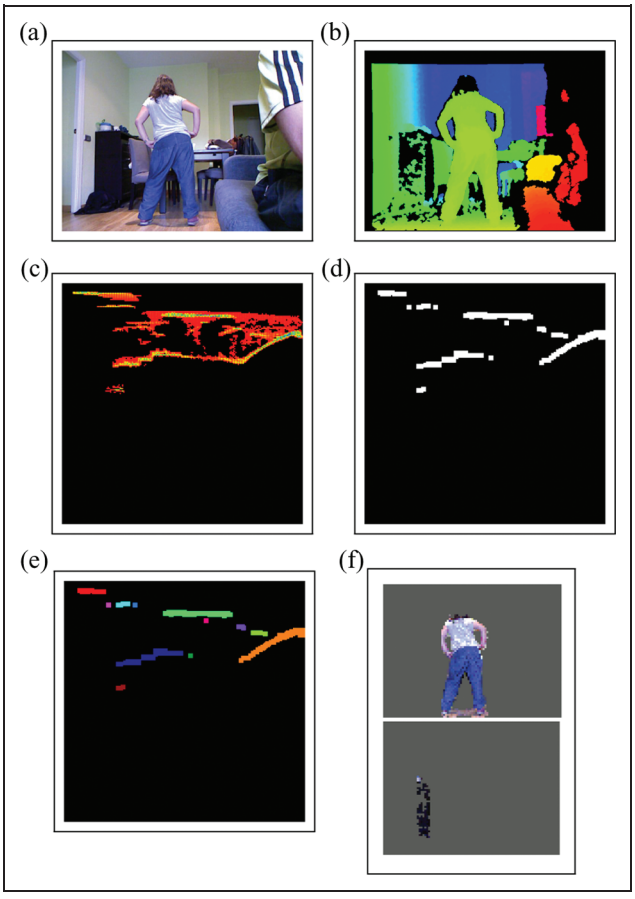

Figure 6. User detection, thanks to polar-perspective maps (PPMs). (a) The input red-green-blue (RGB) image given by the range-imaging device (kinect). (b) The depth map, as supplied by the kinect, and remapped to visible colors (Hue scale). (c): The PPM, remapped to visible colors (Hue scale). Empty bins are shown as black pixels, red colors correspond to almost empty bins, while green bins correspond to fuller bins and blue the fullest. (d) The previous PPM, maintained at a threshold and thus transformed into a binary map. (e) The connected components labeled by color of the PPM set at a threshold. (f) The remaining connected components after eliminating the ones that do not pass the tests on the bounding boxes. On the top image, the person is a true positive. Note that the cupboard in the bottom image is incorrectly identified as a person.

The full processing pipeline is visible in Figure 5, and some samples are shown in Figure 6.

Each pixel of the point cloud is transformed into the PPM coordinates and the corresponding cell of the PPM is aggregated. We maintain at the same time a *reverse map* that stores the list of 3-D points that were projected onto a given cell of the PPM. A sample is visible in Figure 6(c). Then the PPM is thresholded: all the cells that have a minimum of pixels are marked as 1, the others as 0, thus generating a binary mask, as visible in Figure 6(d).

The objects in the scene correspond to the connected components in the image. We use a lightweight, fast algorithm for fetching the different components of the PPM,[31] as visible in Figure 6(e). The 3-D bounding box of each connected component is obtained by considering the cluster made by all the 3-D points that belong to that component, which is done using the previously mentioned reverse map.

Then, we focus on detecting standing humans: some empirical thresholds on the bounding boxes will discard most of the connected components, enabling to keep only the ones that are shaped as a standing person. In our implementation, the width must be between 0.3 and 1 m, the height between 1.2 and 2.1 m (thus also including most children aged over 6), and the depth between 0 and 1 m. These values were chosen so that they include all human configurations, while discarding a reasonable amount of nonhuman clusters. Bounding boxes with dimensions out of these ranges are discarded. In other words, we aim at having a false negative rate of 0, while having a low false positive rate. As can be seen in the sample in Figure 6(f), we indeed have some false negatives from time to time. All the objects in the scene that have dimensions in these spans will indeed be labeled as users.

Finally, the 3-D clusters that passed all tests are likely to be users. We then shape a PPL message for each cluster. The user mask corresponds to the 2-D reprojection of the cluster in the camera frame. In this PPL are also stored the corresponding parts of the RGB and depth images and the 3-D COM of the centroid.

Tabletop PPLP

The detector presented in this section also handles depth information, and the erroneous recognition of remote objects

is limited using a so-called depth *clamping*: the depth values greater than a given threshold are not taken into account.

People standing on the floor generate a point cloud that is similar to objects standing on a tabletop. Detecting objects on a planar surface such as a tabletop is a problem that has already been tackled by other authors, most notably for object grasping.[32,33] We here chose to combine some of these techniques in an innovative way to find the users in front of the robot.

We implemented a PPLP that detects the ground plane, thanks to a statistical method, and then retrieves the pixel blobs of the objects being on top of that plane. We here make the strong hypothesis than all the blobs visible in a given range of distances are only users. For instance, no furniture, box, or any object should enter in this range of distances. This is a very restrictive hypothesis, but it can be met in some cases: an uncluttered environment, such as a wide living room or a lab, can fulfill this condition. The pipeline is illustrated in Figure 7, and a sample is visible in Figure 8.

We suppose the users need to be detected only in a limited and static range of distances. This makes sense in a configuration where the camera is in a given position, say a fixed camera on a robot with limited motion capabilities (e.g. the second robot from the right in Figure 1). We name $d_{min}, d_{max} \in \mathbb{R}$ the minimum and maximum distances where the user can appear. Parameter d_{min} would typically be around half a meter, and d_{max} slightly smaller than the closest wall. Then we clamp our depth map: we mark all pixels with a value smaller than d_{min} or greater than d_{max} as not valid. All pixels in $[d_{min}, d_{max}]$ belong either to users or

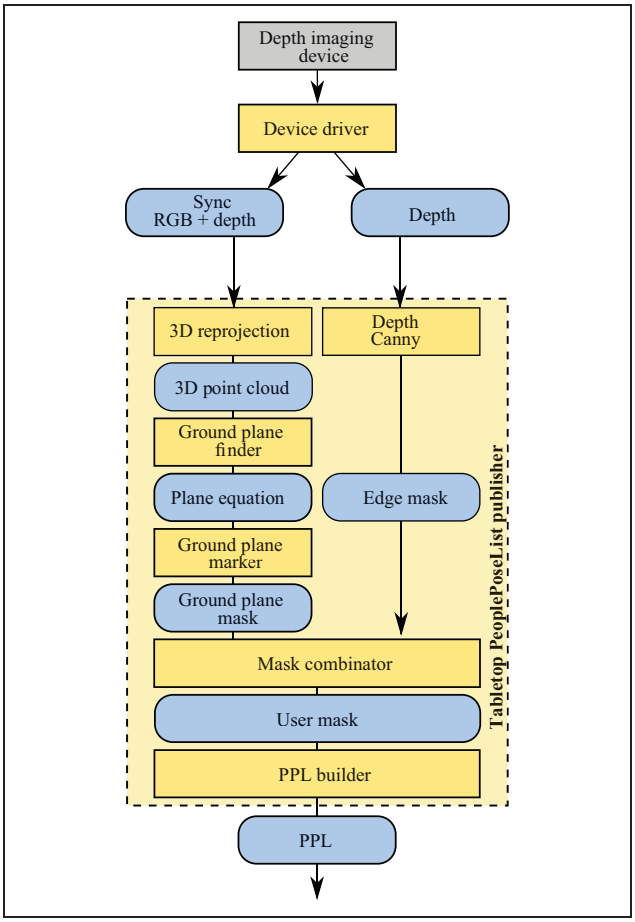

Figure 7. The tabletop user detector pipeline. The caption is identical to Figure 5.

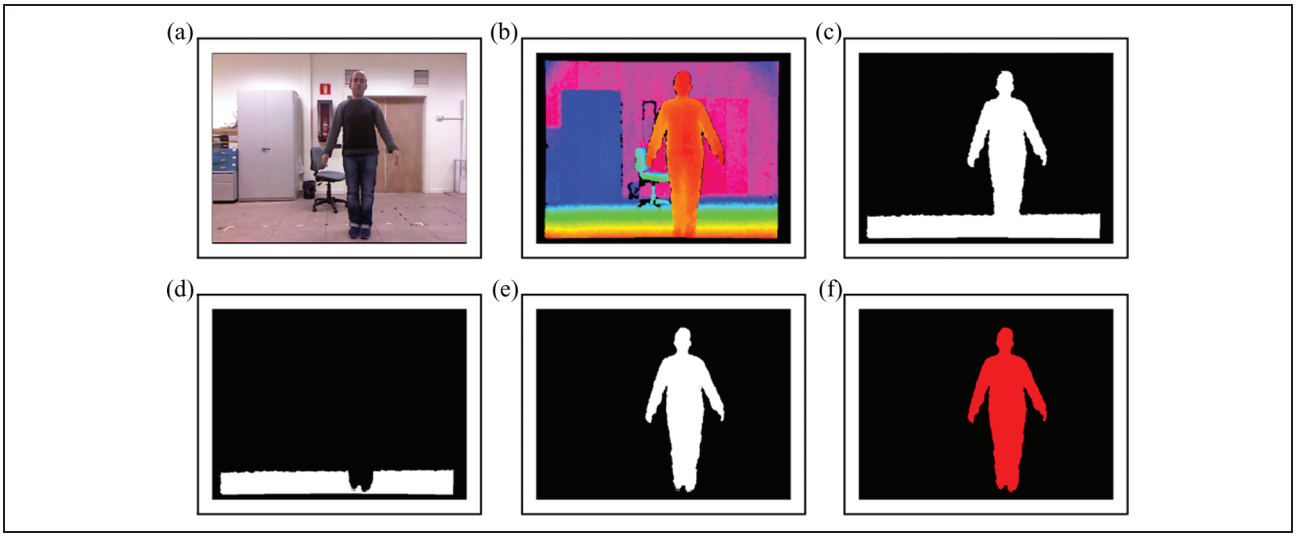

Figure 8. User detection, thanks to the tabletop PeoplePoseList Publisher. (a) The input red-green-blue (RGB) image given by the range-imaging device (kinect). (b) The depth map, as supplied by the kinect, and remapped to visible colors (Hue scale). (c) The valid measures of the depth map, belonging to the chosen range of depth [1,4] (meters). (d) The mask of the pixels belonging to the computed ground plane. (e) The mask of all visible objects. (f) The person blobs. In this example, there is only one person, so one color blob.

to the ground. See how such a filter affects the depth map in Figure 8(c) .

Once the depth map is clamp, the first step is the detection of the ground plane. In other words, we want to detect the points in the image that belong to the ground and that belong to objects on top of it. We used the RANSAC algorithm to compute the plane equation[34] implemented in the PCL.[35] This method consists of estimating the equation of the ground plane and then obtain a binary mask of all the points of the depth image that belong to the ground. The depth image is reprojected to 3-D, giving us a 3-D point cloud of the ground. If the ground is visible in the depth image, then statistically, a great amount of the 3-D points belong to that ground. However, some 3-D points do not belong to the ground, such as object pixels. Statistically speaking, most 3-D points are *inliers* for the ground plane equation, while some are *outliers*. The ground plane equation can thus be restored with a statistical algorithm operating on the point cloud and capable of discarding outliers while optimizing the equation for the inliers.

Once we obtain the equation of the ground, we can evaluate which pixels of the depth map belong to it. For each pixel, we evaluate the absolute distance between its 3-D reprojection and the plane. If the distance is greater than a given threshold, the point is evaluated as belonging to an object. Otherwise, it belongs to the ground. This enables the generation of a ground plane mask, as seen in Figure 8(d).

The user mask is obtained by combining the binary inverse of this first mask (thus removing all ground points) with the depth mask of the points in $[d_{min}, d_{max}]$ described before. In order to separate aligned users, we combine this user mask with a Canny filter on the depth map that will mark the edges (depth disparities) as described previously. This generates a binary map where objects are indicated by a positive value, the rest being 0. A sample is visible in Figure 8(e).

In order to quickly retrieve the connected components in the binary map, we apply the same method used by the PPM-based PPLP (subsection on the PPM-based PPLP). We suppose that all objects in $[d_{min}, d_{max}]$ can only be users. Consequently, all connected components obtained from the binary map correspond to users. A sample is visible in Figure 8(f). These components are then passed to a PPLP that shares this message with the rest of the system.

Benchmarking people detectors

In order to evaluate the performance of the implemented people detectors, we developed a benchmark application, that takes advantage of the PPL message, the common message type published by all PPLPs introduced in the section on defining a common interface for people detectors.

The benchmark application uses files containing both the input data (RGB and depth images) and the manually labeled user position (user mask). The input data are passed to the different PPLPs and the output is compared with the provided user masks.

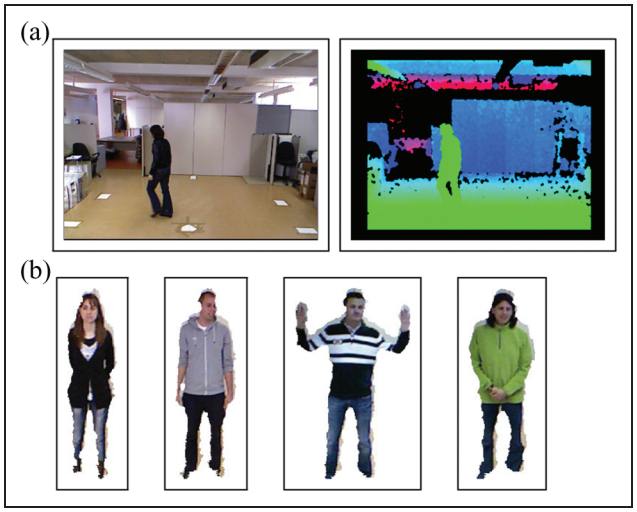

Figure 9. Some samples of the DGait data set. (a) Sample image from the videos of a user. Depth image has been remapped to visible colors. (b) Example of the diversity of users in the data set.

We evaluated the performance of the PPLPs with two data sets that provide the input data: the public academic DGait database and RoboticsLab People Dataset (RLPD). The input data contain synchronized, labeled, visual, and depth information.

These data sets present conditions comparable to the HRI scenarios where our robots are applied, that is, daily environments such as homes, schools, or day care centers. These are indoor spaces where natural and artificial lightings can be combined.

The public academic *DGait database*[36] contains video sequences of 55 users, both female and male, walking on a stage with varying light conditions. Some samples of this data set are visible in Figure 9.

The RLPD is a new homemade data set containing real data acquired from one of our social robots in the laboratory.[37] The actors used to collect data for this data set were engaged in a realistic HRI scenario in which one or several users interact naturally with the robot: addressing the robot, using gestures, respecting the proxemics distance, and so on. The ground truth user positions have been manually labeled in each of the 600+ frames. Some samples are visible in Figure 10.

This data set is licensed under the terms of the GNU General Public License version 2 as published by the Free Software Foundation, and freely available to download on the author's website, along with images and videos (https://sites.google.com/site/rameyarnaud/research/phd/robotics-lab-people-database).

Benchmarking

The benchmark of a PPLP is then made by running it on each frame of all video sequences of the databases and comparing its results with the ground truth.

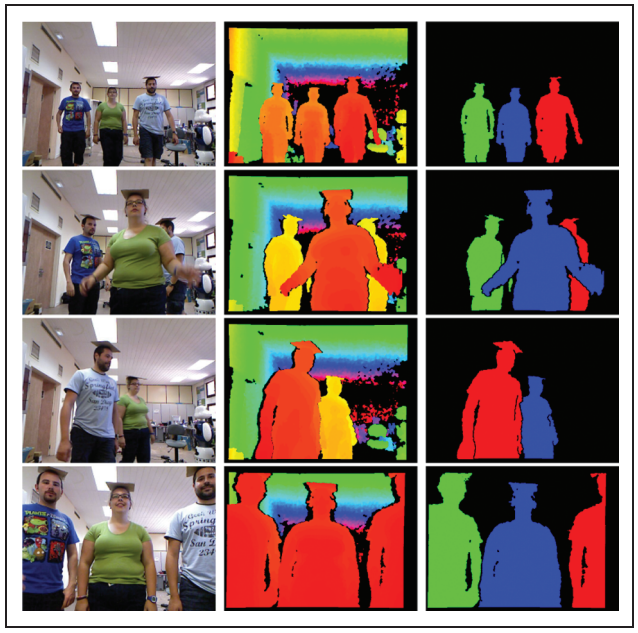

Figure 10. Some samples of the RoboticsLab People Dataset (RLPD). From left to right: column 1: the red-green-blue (RGB) image, column 2: the depth image; column 3: the manually labeled user map. The data set gathers some challenging features: partial (second row) or complete occlusions (third row), user not fully visible (fourth row). Note how the manual color indexing of the users is consistent (third column): the same user always corresponds to the same color.

Considering the metrics defined by Olson and Delen,[38] we computed the true positive, true negative, false positive, and false negative rates. *Positive* means that a person is detected in the current input, while *negative* indicates that no person is detected. *True* and *false* describes if this result is correct or not.

With all these values, we calculated the *accuracy* and the *hit rate* of each algorithm. Accuracy refers to the percentage of correctly evaluated frames:

$$\text{Accuracy} = \frac{\text{true positive} + \text{true negative}}{\text{total number of frames}} \times 100$$

In the case of the hit rate, it measures the number of people successfully detected, considering the frames that certainly contain al least one person:

$$\text{Hit rate} = \frac{\text{true positive}}{\text{total number of frames with people}} \times 100$$

Results

The performance of the different 3-D people detectors was measured, thanks to the benchmark based on the already mentioned DGait database and RLPD. The results of the former are included in the subsection on benchmarking and subsection on results with the RLPD5.2.2 presents the results of the latter.

Results with the DGait database. The results obtained when using the DGait database are gathered in Table 1. In the case of the face-based PPLP (first column), the false positive rate is very low. This is due to the use of the depth information to discard false positives, it limits the detection of nonexisting faces. However, the hit rate is under 25%, that is, less than one-fourth of the users were detected.

If we focus on the HOG-based PPLP, we observe that the accuracy of this algorithm is roughly 70%. If we consider the fact that the detections are made using only the color information, the depth being used only for false positive removals, this is a fairly reliable algorithm for the conditions of the DGait data set: more than two-thirds of the users will be detected.

We can be surprised by the great results obtained by the NiTE-based PPLP. However, seeing that the user mask supplied by the data set was actually obtained by its authors using the NiTE middleware, we can easily understand this results. In other words, the ground truth used to evaluate our PPLP was created with same middleware we used in the NiTE-based PPLP. For this reason, its accuracy and precision are roughly equal to 1. In fact, this PPLP measures the accuracy of our wrapper, that is, the common interface presented in the section on defining a common interface for people detectors, that converts the three image topics (RGB, depth, and user multimask) to PPL. Note that this does not mean that the NiTE-based PPLP never fails, it means that it fails as often as the labeling of the data set: the labeling made by NiTE is faithfully converted by our wrapper, and so the failure in the labeling, triggered by NiTE failures.

The PPM-based PPLP has an accuracy of 65.8%. It is designed to work specifically with standing humans, which is no limitation for this benchmark, as all users are standing or walking in this data set. Note that because the PPM has no maximum distance, all the remote objects (walls, cupboards, etc.) are used for its building and are erroneously recognized as users from time to time. This explains the numerical difference between the hit rate and accuracy for this detector.

The tabletop-based PPLP has an accuracy very close to 1. This means that the detector is almost always true in its predictions. We have to remember the strong assumption we made before though: we consider an uncluttered environment, with users in a range of distances. The database is made of users walking on a stage, which corresponds exactly to these settings. This justifies the good results. This accuracy would be much lower if there was an alien object on stage, such as a cupboard for instance.

Results with the RLPD. The RLPD offers a high level of complexity: the occlusions are frequent, the users are sometimes partially shown in the images, and in general, they are somewhat far away from the camera. The results obtained with this data set are summarized in Table 2.

The NiTE PPLP has a high accuracy and hit rate (both above 90%), which confirm the good performance of the

Table 1. Benchmark results for the PPLPs with the DGait database.

PPLPs	Face detection	HOG based	NiTE based	PPM based	Tabletop based
True positives	3655	11,415	16,076	12,719	16,020
True negatives	1606	1422	1608	1161	1586
False positives	36	846	9	3857	11
False negatives	12,457	4713	1	3353	45
Accuracy (%)	29.6	69.8	99.9	65.8	99.7
Hit rate (%)	22.7	70.8	100.0	79.1	99.7

HOG: histogram of oriented gradients; PPLP: PeoplePoseList Publishers.

Table 2. Benchmark results for the PPLPs with the RLPD.

PPLPs	Face detection	HOG based	NiTE based	PPM based	Tabletop based
True positives	259	66	1255	166	581
True negatives	63	63	63	62	0
False positives	0	0	3	4	63
False negatives	1131	1322	130	1222	807
Accuracy (%)	22.2	8.89	90.8	15.7	40
Hit rate (%)	18.6	4.76	90.6	12	41.9

HOG: histogram of oriented gradients; PPLP: PeoplePoseList Publishers; RLPD: RoboticsLab People Dataset.

algorithm that was already seen with the DGait database (both metrics above 99%).

The tabletop PPLP had a very good performance on the DGait data set, as it had over 99%. However, on this benchmark, its performance is lower: it has a detection rate of around 40%. Indeed, as it can be seen in the sample images (Figure 10), the ground is hardly visible in the images acquired by the camera, which generates a poor estimation of the ground plane. We can then conclude that the tabletop PPLP is not appropriate for the robot spatial configuration used in the RLPD.

In a way similar to the tabletop PPLP, the PPM PPLP performs much worse on our homemade data set than on the DGait data set: its accuracy lowers from 66% to 16%. The reasons are very similar: the limited visibility of the ground provokes a poor estimation of the transform between the camera space and the perspective map, which creates an incorrect projection into the latter.

On the other hand, the face detection PPLP has low accuracy and hit rate, but similar to DGait (around 25%): like the other database, the users are most of the time several meters away from the robot and often turn their back to the camera, which are challenging conditions for face detection.

Finally, the HOG detector is the most curious: its detection and hit rates, which were above 70%, fall drastically to under 10%. As it can be seen in the sample images, the users are never fully seen, and especially their legs are most of the time out of the picture frame: the HOG detector being trained for detecting fully visible pedestrians, it performs very poorly on the RLPD.

The comparative performance of the different PPLPs with both data sets is clearly presented in Figure 11.

Discussion

The previous section has presented the results of five different PPLPs, that is five people detection algorithms using a common data structure, employing RGB-D data. These results were obtained using all the frames of two databases of videos where standing people walk around. Consequently, the performance of these PPLPs in other conditions (e.g. people sit or lying in a bed) is not clear. Some PPLPs, such as the PPM, detect standing humans and rely on general features of standing people, for instance, the height. These people detectors will not work properly under different situations. However, other PPLPs with lower accuracy and hit rate will perform better in more heterogeneous conditions. For example, the face detection PPLP obtained the worst results. This poor performance is due to the fact that the users visible in the video stand several meters away from the camera and are often turning her back to it. These condition does not correspond to the domain of use of face detection, thought for interaction at short distance. If a user is sitting at a table few meters in front of the robot, its performance will certainly increase. We already commented that the face detection-based PPLP only works well with frontal faces. The data set is made of users walking on a stage, which means that, unless they look in direction of the camera, their face is not visible. Furthermore, this detector only detects well faces that are reasonably large in the input image (the minimum size can be of course lowered, but this triggers an important increase of false positives). Both these reasons account for the low hit rate.

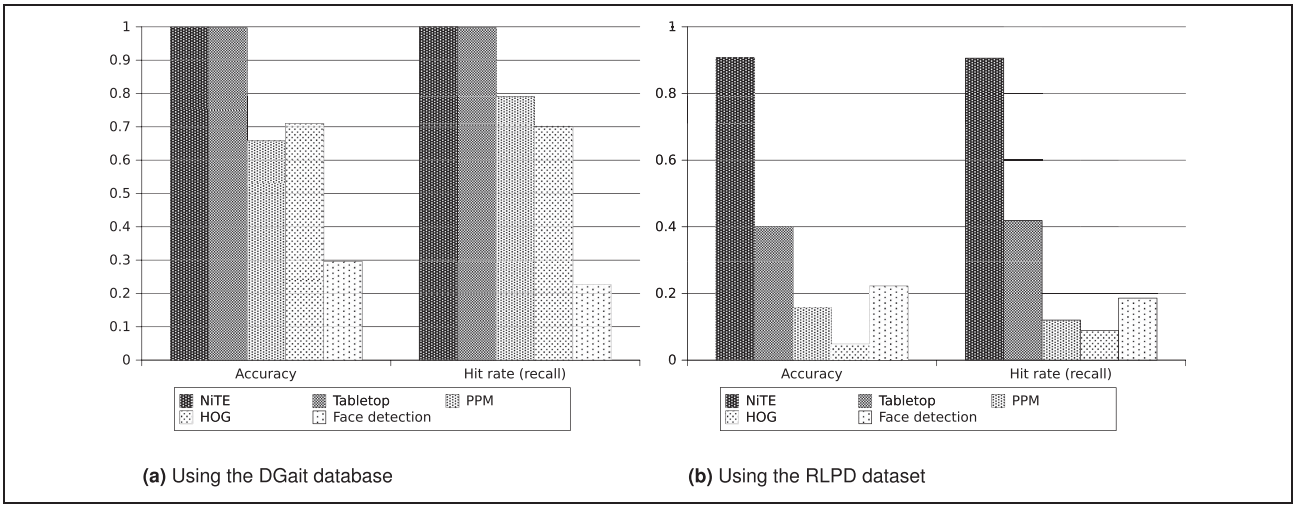

Figure 11. Comparative performance of the different PeoplePoseList Publishers.

The PPM PPLP presents a quite good accuracy (two-thirds) with the DGait database. This algorithm does not use a depth range to detect people, resulting in detecting remote objects (walls, cupboards, etc.) as people. The depth clamping of the tabletop detector prevents this confusion and this is one of the reasons of its great performance. This technique, which apparently is very restricted, improves the results substantially. However, as shown in the results obtained with the RLPD, when the ground is hardly visible, the performance of these two PPLPs decreases dramatically. Consequently, they can be successfully applied to SRs whenever the ground where the users stand is visible.

The uncluttered environment is a key explanation for the outstanding performance of segmentation-based detectors (the NiTE and the HOG PPLPs) in the DGait database. The introduction of any object shaped like a human user, say a coat rack, would generate false positives and thus alter the accuracy rate of these. The algorithms based on the color analysis (HOG and face detection) would not be affected by this new object. The use of a more challenging data set (RLPD) reveals the weakness of these people detectors when the users are not fully visible and occlusions exist. Under this situation their performance is reduced, in particular the HOG detector becomes inapplicable on the RLPD.

For this reason and considering that it performs well when the user is fully visible (even with people who are not facing the robot), the HOG algorithm can be applied to social robots in limited situations.

The results obtained by the NiTE-based PPLP are remarkable. The two benchmarks confirm that the NiTE-based PPLP is a useful method for detecting users in a social robot, even in challenging conditions. As said before, the users need to move to start their detection and tracking. The NiTE middleware is aimed at playing video games on the Microsoft XBox 360, and as human players are bound to move their body to play, the need for motion is not a problem in this context. For HRI though, detecting a still, motionless audience becomes challenging. This becomes an issue of paramount importance if the user stands still in front of the robot while speaking with it. Furthermore, if the sensor is mounted on a mobile robot, people detection will be performed while having the roboot in motion. In such cases, there is no longer any static background and the segmentation performed by the NiTE-based PPLP is bound to fail. Wrong segmentations also happens when the sensor is still. For instance, in Figure 4(c), the leg of one user is erroneously labeled as belonging to the other user, as it must appear geometrically closer to the latter than to the former. Similarly, when users stand close to walls, pieces of wall are frequently added to their user mask.

All the PPLPs presented in this work consider depth data. Using this kind of data, unlike other algorithms, our PPLPs will not be fooled by everyday objects, such as photos of people.

Thanks to the common interface presented in the section on defining a common interface for people detectors, the different people detectors can be applied to different robots smoothly. However, depending on the robot configurations, some people detectors are more suitable than others. For instance, in the case of *Moppy* (the second robot from the left in Figure 1), the tabletop detector will perform better than the face detection-based PPLP. This is because Moppy is a car-like robot with a depth sensor located close to the ground and, thus, faces are very difficult to detect but the floor is easily perceived.

Conclusions

In this article, we tackled the problem of detecting users around a social robot in HRI scenarios. In these scenarios, usually interaction is conducted at short distances and robot displacements are limited to approximating a potential user. Moreover, in order to achieve a natural HRI, it is crucial to obtain the distance to the user using the sensors

on board the robot, so the environment does not have to be altered. Depth images ease the development of people detectors that fulfill these requirements.

In the same manner animals employ color and depth information to perceive its surroundings, in this work, we have presented four novel people detectors using color and depth images. On the one hand, the face detection-based and the HOG-based detectors use a seed pixel and run a propagation algorithm using 3-D data to detect possible human heads and bodies, respectively. Then, the candidates are filtered based on geometrical constraints. On the other hand, the PPM-based and the tabletop detectors have been adapted to the domain of HRI. The first one is based on the fact that a person appears as a set of points tightly close one to another, forming a 3-D blob with particular dimensions. The last one, the tabletop detector, considers people as objects standing on a planar surface (the ground) within a range of distances. Besides, NiTE-based people detector works with the kinect official API. Its widespread use made us to include it to show how to integrate a well-known people detector and for testing purposes. The NiTE-based PPLP supplies a fairly robust detection and tracking of the users, but that is only appropriate for a static camera configuration and requires the users to move to detect them.

All people detectors have been integrated in our robotic control architecture, standardizing their output by means of a common data structure called PPL. This results in numerous advantages: programming language abstraction (thanks to ROS msg mechanisms), workload redistribution between several computers, agile integration of new algorithms, debugging and benchmarking made easier, and use of common visualization tools.

All algorithms were carefully tested and benchmarked. Some algorithms turned out to perform notably better than others and all of them have different domains of use; consequently, each of the different PPLPs has both strengths and limitations, and the experimental setting will help choosing a configuration or another. A clever fusion system would then take advantage of each detector when it is the most relevant and discard its output when it is out of its domain of use. This point will be developed in future works.

To easy the verification, replication, and extension of this work, all the source codes used in this work are available online under an open source license (https://github.com/UC3MSocialRobots https://github.com/UC3MSocialRobots).

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: The research leading to these results has received funding from the projects Development of social robots to help seniors with cognitive impairment (ROBSEN), funded by the Ministerio de Economia y Competitividad, and RoboCity2030-III-CM, funded by Comunidad de Madrid and cofunded by Structural Funds of the EU.

References

1. Jung J, Dan B and An K. Real-time human tracking using fusion sensor for home security robot. *Consumer Electronics (ICCE), 2012 IEEE International Conference on* 2012, Las Vegas, NV, USA, 1 March 2012.
2. Fernández-Caballero A, López MT, Castillo JC, et al. Real-time accumulative computation motion detectors. *Sensors* 2009; 9(12): 10044–10065.
3. Berliner T and Hendel Z. Modeling of humanoid forms from depth maps. Patent application 20100034457, USA, 2007.
4. Kepski M and Kwolek B. Human fall detection by mean shift combined with depth connected components. *Comput Vision Graphics* 2012; 7594: 457–464. Lecture Notes in Computer Science, Springer, Berlin: Heidelberg.
5. Ramey A, Gonzalez-Pacheco V and Salichs MA. Integration of a low-cost RGB-D sensor in a social robot for gesture recognition. In: Aude Billard, Peter Kahn, Julie A. Adams and Trafton J. Gregory (eds) *Proceedings of the 6th International Conference on Human Robot Interaction. HRI '11*, New York, NY, USA, 8–11 March 2011, ISBN 978-1-4503-0561-7, pp. 229–230. New York: ACM.
6. Bellotto N and Hu H. Multisensor-based human detection and tracking for mobile service robots. *Syst Man Cybernet Part B* 2009; 39(1): 167–181.
7. Ramey A, Malfaz M and Salichs MA. Fast 3D cluster-tracking for a mobile robot using 2D techniques on depth images. *Cybernet Syst* 2013; 44(4): 325–350.
8. Leite I, Castellano G, Pereira A, et al. Modelling empathic behaviour in a robotic game companion for children. In: *Proceedings of the Seventh Annual ACM/IEEE International Conference on Human-Robot Interaction - HRI '12*. New York, NY, USA, 5–8 March 2012, ISBN 9781450310635, p. 367, New York: ACM Press.
9. Lambert D. *Body language*. London: Harper Collins, 2004.
10. Walters M, Dautenhahn K, te Boekhorst R, et al. An empirical framework for human-robot proxemics. In: *Proceedings of the Symposium New Frontiers in Human-Robot Interaction. A symposium at the AISB 2009 Convention* Heriot-Watt University, Edinburgh, Scotland, 6–9 April 2009, ISBN -190295680X, pp. 144–149, http://www.aisb.org.uk/.
11. Susperregi L, Martínez-Otzeta JM, Ansuategui A, et al. Rgb- d, laser and thermal sensor fusion for people following in a mobile robot. *Int J Adv Robot Syst* 2013; 10: 271.
12. Dudek G, Sattar J and Xu AXA. A Visual Language for Robot Control and Programming: A Human-Interface Study. *Proceedings 2007 IEEE International Conference on Robotics and Automation* 2007.
13. Tomé D, Monti F, Baroffio L, et al. Deep convolutional neural networks for pedestrian detection. *Elsevier J Signal Process* 2016; 47: 482–489.

14. Giusti A, Ciresan DC, Masci J, et al. Fast image scanning with deep max-pooling convolutional neural networks. In: *IEEE International Conference on Image Processing, ICIP 2013*, Melbourne, Australia, September 15–18, 2013, pp. 4034–4038, Melbourne: IEEE.
15. Girshick RB, Donahue J, Darrell T, et al. Rich feature hierarchies for accurate object detection and semantic segmentation. In: *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014*, Columbus, OH, USA, June 23–28 2014, pp. 580–587. Columbus: IEEE Computer Society.
16. Hosang JH, Omran M, Benenson R, et al. Taking a deeper look at pedestrians. In: *CoRR 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, MA, USA, 7–12 June 2015, abs/1501.05790. IEEE Computer Society.
17. Viola P and Jones M. Robust real-time face detection. *Int J Comput Vision* 2004; 57(2): 137–154.
18. Viola P and Jones M. Rapid object detection using a boosted cascade of simple features. In: Anne Jacobs and Thomas Baldwin (eds) *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 8–14 December 2001, volume 1. Los Alamitos, CA, USA: IEEE Computer Society ISBN 0-76951272-0, pp. 511–518. Los Alamitos: IEEE Computer Society.
19. Lienhart R and Maydt J. An extended set of haar-like features for rapid object detection. In: *Image Processing 2002 Proceedings 2002 International Conference on*, Rochester, New York, USA, 22–25 September 2002, IEEE, ISBN 0-7803-7622-6.
20. Rowley H, Baluja S and Kanade T. Neural network-based face detection. *Pattern Anal Machine Intell IEEE Trans* 1998; 20(1): 23–38.
21. Rehg J.Vision-based speaker detection using bayesian networks. In: *Computer Vision and Pattern Recognition, 1999 IEEE Computer Society Conference on* 1999, 23–25 June 1999, Ft. Collins, CO, USA. IEEE Computer Society, ISBN 0-7695-0149-4.
22. Dalal N and Triggs B. Histograms of oriented gradients for human detection. In: *Computer Vision and Pattern Recognition, 2005 CVPR 2005 IEEE Computer Society Conference on (Volume:1)*, 20–26 June 2005, San Diego, CA, USA. IEEE Computer Society, ISBN 0-7695-2372-2.
23. Fong C, Dotson R and Bejczy A. Distributed microcomputer control system for advanced teleoperation. In: Antal KB and Rajan S (eds) *Robotics and Automation Proceedings. 1986 IEEE International Conference on volume 3*, 7–10 April 1988, San Francisco, California, USA: IEEE Computer Society. pp. 987–995.
24. Zhu Q and Yeh M. Fast human detection using a cascade of histograms of oriented gradients. *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on* 17–22 June 2006. New York, NY, USA: IEEE Computer Society 2006, ISBN 0-7695-2597-0.
25. Pang Y, Yuan Y, Li X, et al. Efficient HOG human detection. *Signal Process* 2011; 91(4): 773–781.
26. Latta S and Tsunoda K. Gesture keyboarding. Patent application 12/391,145, USA, 2009.
27. Howard A, Matthies LH, Huertas A, et al. Detecting pedestrians with stereo vision: safe operation of autonomous ground vehicles in dynamic environments. In Kaneko, Makoto; Nakamura Y (ed.) In: *Proceedings of the 13th Int. Symp. of Robotics Research*, 26–29 November 2007. Hiroshima, Japan: Springer-Verlag Berlin Heidelberg.
28. Quigley M, Gerkey B, Conley K, et al. ROS: an open-source Robot Operating System. In: *ICRA Workshop on Open Source Software*, 17 May 2009. Kobe, Japan.
29. Bradski G and Kaehler A. *Learning OpenCV: computer vision with the OpenCV library*. 2008. O'Reilly Media, Inc., Gravenstein Highway North, Sebastopol, CA.
30. Canny J. A Computational Approach to Edge Detection. *IEEE Trans Pattern Anal Mach Intell* 1986; PAMI-8(6): 679–698.
31. Ramey A. Playzones: a robust detector of game boards for playing visual games with robots. In: Ollero A, Martinez de Dios JR, Cobano JA, Ferruz J, Caballero F, Merino L and Rodriguez-Castaño A (eds) *Libro de Actas del 3er Workshop ROBOT'11: Robotica Experimental*, chapter 7c. Sevilla: ROBOT'11, 28–29 November 2011. pp. 626–638.
32. Blodow N and Rusu R. Partial view modeling and validation in 3D laser scans for grasping. In: *Humanoid Robots*, Paris, France, December 2009.
33. Marton ZC, Goron L, Rusu RB, et al. Reconstruction and verification of 3D object models for grasping. In: Pradalier C, Siegwart R and Hirzinger G (eds) *Robotics Research* 2011. Springer Tracts in Advanced Robotics, vol 70. Springer, Berlin, Heidelberg.
34. Fischler MA and Bolles RC. Random sample consensus: a paradigm for model Fitting with applications to image analysis and automated cartography. *Commun ACM* 1981; 24: 381–395.
35. Rusu RB and Cousins S. 3D is here: point Cloud Library (PCL). In: Zexiang Li and Yuan F. Zheng (eds) *IEEE International Conference on Robotics and Automation* 9–13 May 2011, 1–4. Shanghai, China: IEEE Computer Society.
36. Igual L, Lapedriza A and Borràs R. Robust Gait-Based Gender Classification using Depth Cameras. In: *Eurasip Journal On Image And Video Processing*, 2013: 1. doi:10.1186/1687-5281-2013-1.
37. Ramey A. *Local user mapping via multi-modal fusion for social robots*. PhD Thesis, University Carlos III of Madrid, Spain, 2014.
38. Olson D and Delen D. *Advanced data mining techniques*. Heidelberg: Springer, 2008.