3D object recognition for anthropomorphic robots performing tracking tasks

S. Satorres Martínez[1] · A. Sánchez García[1] · E. Estévez Estévez[1] · J. Gómez Ortega[1] · J. Gámez García[1]

Abstract
Object recognition is still a major research issue of particular relevance in robotics. The new trend in industrial and mainly in service robotics is to perform manipulation tasks in an unstructured environment working in synergy with humans. To perform tasks in an environment that is not perfectly controlled, robots need adequate perceptual capabilities. Among various types of sensors available for robotic systems, the time-of-flight (ToF) camera is one of the most utilized since it simultaneously provides intensity and depth data at a high frame rate. Our proposal makes use of this technology exploiting both, depth and grey-scale information. Therefore, intensity and geometric features are fused together to allow 3D object recognition in real scenes in presence of partial occlusions. As a case study, an object tracking task for an anthropomorphic robot is presented. Experimental results demonstrate the effectiveness of the proposed method, not only providing reliable visual information about the object to be tracked but also recognizing potential obstacles which should be avoided during the robot arm movement.

Keywords 3D vision · Feature fusion · Object recognition · Tracking tasks · Anthropomorphic robots

1 Introduction

New robotics applications face with tasks in which human operators and robots work in a collaborative manner sharing the same workspace. This synergy between humans and robots enables the development of more flexible and complex tasks but also requires a precise information about the relative position between the robot and the objects of interest [1]. Because of this, the problem of tracking objects and avoiding obstacles has received a good deal of attention [2].

Object tracking task requires a previous object recognition. In this sense, computer vision is a feasible technology as it can provide accurate and real-time information of the 3D scene. The term '*object recognition*' comprises two concepts: *identification* and *location* [3]. Object *identification* is the determination of which 3D objects are present in a real-world scene, and *location* involves the estimation of the pose, that is position and orientation, of the identified objects with respect to some global coordinate system attached to the scene.

Object recognition in real scenes is a challenging task [4]. The starting point is the input image data that may be obtained from one or more sensors such as CCD cameras and/or those that deliver three-dimensional imaging. Image processing systems based on CCD or CMOS cameras are sensitive to illumination, shadows, scale and pose. New devices such as time-of-flight (ToF) cameras do not suffer from this limitation delivering intensity and range data at a high frame rate. Nevertheless, they are affected by noise and are low resolution compared with other depth map estimation systems [5].

Once sensed data from the environment are available, objects have to be extracted from the background, modelled by a set of features, and finally recognized. Numerous approaches have been attempted over the years in each of these research areas and the problem has been solved, to some extent, in unoccluded scenes under controlled illumination conditions [6, 7].

One of the main drawbacks is that real scenes may often be ambiguous. For instance, an object may look different while two distinct items with similar shape, but differing in size, could look the same, depending on the point of

Publisher version: https://doi.org/10.1007/s00170-019-04058-6

view [8]. Due to the large amounts of sensor information and high variation in objects of the same type, object recognition is a high dimensional problem [9]. Despite these difficulties, objects of the same type do share some common characteristics, or can be used for a specific task, and even are typically located at certain positions. Using these insights, the dimensionality of the problem is reduced, and object recognition becomes more tractable.

Our main objective in this study is to identify and estimate poses of an object of interest and integrate this information into a robot vision system to perform tracking tasks. The following three key components are required to achieve the objective sought:

- Selection of the image acquisition device. As the robot should be able to adapt its behaviour in real time, recognition cannot take more than a fraction of a second. This puts constraints on computation, and what is more important, precludes the use of slow 3D acquisition systems. For this reason, a time-of-flight camera has been used to provide depth and grey-scale information.
- Image processing. To reduce the dimensionality of the problem, and therefore image processing time, prior knowledge of the scene can be utilized. It is assumed that the object of interest lies on a surface which should be avoided during the robot movement. Hence, the image processing proposal has taken into account the former assumption.
- Validation. In the majority of the studies, the validation of the 3D object recognition algorithms is done using 3D model data bases [7]. In this work, the detection accuracy and the processing speed of the computer vision algorithms have been validated in a robotic platform. This required the development of a control strategy to move the robot arm. Authors in previous work have developed a predictive control strategy [10, 11], to assure that the robot arm follows, at a certain distance, the reference position, that is, the target center of gravity.

Therefore, the main contribution of this paper is proposing a method to identify and locate a target and potential obstacles fusing features extracted from different sources, three-dimensional and grey-scale images. The method starts with reducing data dimensionality. To achieve this, the three-dimensional image (point cloud) is processed following a sequential procedure removing, first, unreachable points and, then, the potential obstacles. The recognition of the target is done in this simplified point cloud. If such recognition fails, the grey-scale image is also processed. Grey-scale features together with depth image features are included in an supervised learning classifier to identify the target. It will be demonstrated that the fusion of different sources of information improves the object recognition.

The rest of the paper is organized as follows. Section 2 presents related work in 3D object recognition and vision-based robotics systems. Section 3 details the technical approach that includes object segmentation algorithms, working on both depth data and grey-scale images, features extraction and fusion. The proposal feasibility is assessed in a real robotic set-up, presented as a case study in Section 4. Finally, the conclusions are drawn in Section 5.

2 Related work

Three-dimensional object recognition in range data is a very active area of research with a plethora of open problems that need further investigation. A critical overview and the next step in the evolution of object recognition algorithms is presented in [12]. Basically, the relevant work can be classified into two main approaches: *passive* and *active*. Focusing on the data acquisition process, the *passive* approach refers to systems which exhibit limited control over it, while the *active* one applies intelligent control strategies.

Most of the classical contributions are included into the *passive* approach and have been concentrated on designing surface shape representations for recognizing specific objects in a database. A widely used surface representation for mobile manipulation is Fast Point Feature Histograms (FPFH) [13]. On the basis of Point Feature Histogram (PFH) [14, 15], which is a robust multi-dimensional feature descriptor representing the local geometry around a point (p) for 3D point cloud datasets, FPFH goes a step further reducing the computation time. Further modifications exist as Global Fast Point Feature Histograms (GFPFH) [16] and Viewpoint Feature Histograms (VPFH) [17] which put their emphasis on object recognition in a more global manner. Another method which also relies on local surface information from point normals in a local neighbourhood is Radius-Based Surface Descriptor (RSD) [18].

Popular descriptors for object recognition tasks are Spin Images [19], 3D Shape Contexts [20] and an improvement of the latter one in terms of accuracy and reduced memory consumption, the Unique Shape Context [21]. However, these approaches are sensitive to sensor noise and require densely sampled data [22]. Other descriptors work only when intensity information is provided for every point of the point cloud which is the case of RIFT descriptor and intensity-domain spin images [23] and chain codes in range images [24]. Recently, a novel local shape descriptor, Signature of Geometric Centroids (SGC), was applied in [25] to find high-quality potential features correspondences. Apart from the previously cited works, a recent review of generic techniques in feature representation for object recognition can be found in [26].

The exploitation of prior knowledge about the object function is being extensively used to improve recognition. Recent works, mainly focused on robotics field, are based on the concept of *affordance* or exploiting function. The main idea involves associating visual features, extracted from the image processing with the function of the object. In [27], the knowledge of how a human being would grasp an object is considered along with visual features to recognize objects of daily use in human living environments. Identifying a set of objects based on data from internal flex sensors has been addressed in [28].

Active control of a vision sensor is included within the *active* approach. It offers a number of benefits such as bringing into the sensor's field of view regions that are hidden due to occlusion; increasing spatial resolution through sensor zoom or observer motion, and dealing with incomplete information to complete a task [12]. An effort in formalizing the problem and motivating the need for active approaches is discussed in [29]. Few works on active 3D object search are available for robotics applications which among them are included in [30] and [31]. In [30], an object active visual search in a 3D environment performed by a humanoid robot is presented, while [31] is not focused on object search but on accurate mapping of unknown environment by means of sensor planning.

Most of the former approaches require high computational needs. However, new mobile robots, drones or miniature robotics cars, with limited on-board computing capabilities, made use of new concept 'The Internet of Robotic Things' [32] to implement object recognition and tracking tasks. In [33], the image processing unit is located in a remote location and [2] only processes some regions of interest instead of the full image to speed up the tracking process. The same pattern is applied in our proposal but, in lieu of defining regions of interest, unwanted information is removed from the point cloud.

3 Technical approach

Figure 1 depicts all the necessary steps for 3D object recognition based on depth and grey-scale images which were acquired by means of a time-of-flight camera. What is really novel in this approach is that features extracted from both images constitute the feature vector used by the classifier to assign the segmented regions to a category. Combining data from two sources improves the recognition performance especially in the presence of partial occlusions or small object deformations (due to object motion). A detailed description of the two main stages of the image processing proposal is provided below.

3.1 Stage 1. Obstacle recognition

The main aim of this stage is to reduce data dimensionality and recognize potential obstacles that should be avoided

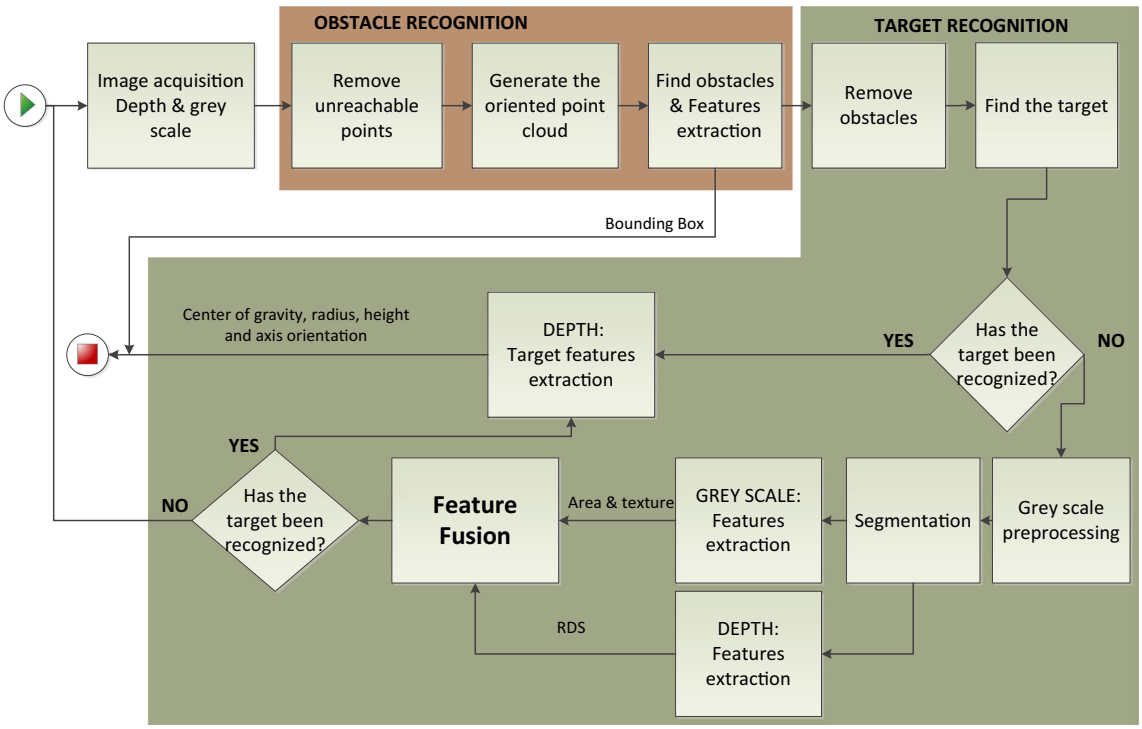


Fig. 1 Overview of the 3D object recognition system

during the task execution. An iterative procedure that first removes unreachable points, then generates an oriented point cloud, and finally recognizes potential obstacles, has been designed.

Unreachable points, not included into the robot workspace, are removed adapting 2D image processing ideas. Thus, a global thresholding algorithm is executed on the depth image to remove points. The global threshold should be the maximum distance reachable by the robot arm. Now, the depth data have been reduced substantially and an accurate but time-consuming algorithm can be applied.

Obstacles and target recognition is based on detecting primitive geometric shapes in the point cloud. Due to its good performance, even if the data set is contaminated by noise, the RANSAC algorithm has been utilized [34]. Input for the RANSAC algorithm is an oriented point cloud and the surface normal vectors are calculated through the following steps:

- Import the point data cloud.
- Build a k-D tree space-partitioning data structure.
- Define the k parameter. This parameter is used to approximate the normal vector. It is a free parameter chosen depending on the sampling resolution and the presence of noise.
- Find the k nearest neighbours of the points.
- Obtain the covariance of the former points.
- Find the eigenvectors.
- Choose the eigenvector with the smallest eigenvalue as surface normal approximation.

Once the input data is available, RANSAC algorithm has to be executed twice: firstly, for identifying and locating planes, and secondly for recognizing cylinders (see next subsection). A plane-shaped surface corresponds to the obstacle. Once the obstacle is recognized a three-dimensional bounding box is calculated. This information is one of the outputs of the object recognition system.

3.2 Stage 2. Target recognition

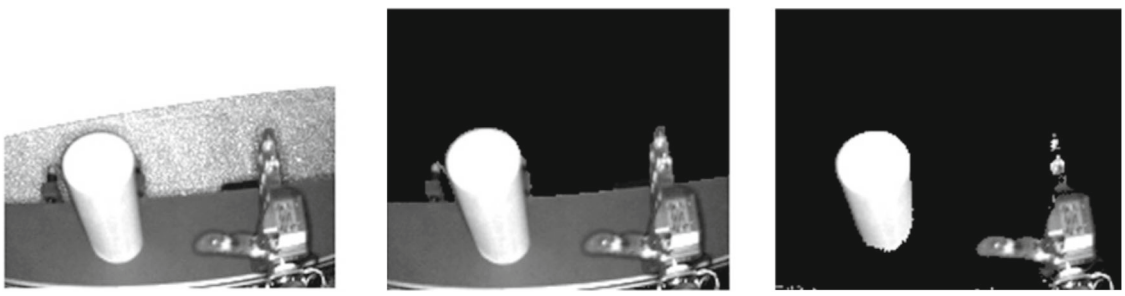
The first step in this stage is to remove the 3D points included in the obstacle bounding box from the depth image. Again, the point cloud data has been substantially reduced. Then, a first attempt to recognize the target is done. For this, the RANSAC algorithm runs for a second time but now for finding the target, that is, cylinders with a predefined radius. If there is a successful detection, the algorithm stops and the target is defined by the following parameters: center of gravity, radius, height and axis orientation. This normally occurs when the target presents a clear cylindrical shape in the depth image and good results are also achieved in presence of partial occlusions. However, depending on the relative position between the ToF and the target, a cylindrical shape cannot be found in the depth image. To solve this, more information needs to be extracted from the images. Next subsections detail the image processing algorithms and the final results in object recognition.

3.2.1 Grey-scale image processing

As it is shown in Fig. 1, the grey-scale image is preprocessed, then utilized together with the depth image for the segmentation step and finally features derived from both images are extracted.

In the grey-scale image preprocessing step, the input data is the grey-scale image (Fig. 2a). Then, points which were removed from the point cloud in the object recognition stage, such as unreachable points or obstacles, are labelled as zero in the grey-scale image (Fig. 2b, c).

Now, objects can be easily extracted from the background based on the grey-level similarity. A region-based method is used for image segmentation. As its name implies, this method groups pixels or subregions into larger regions based on a predefined criteria of growth [35]. Starting with a set of 'seed' points and setting them in each of the remaining



(a) (b) (c)

Fig. 2 Grey scale input data. **a** Grey scale image; **b** Grey scale image after removing unreachable points; **c** Grey scale image after removing obstacles

clusters in the depth image, the algorithm partitions the whole data into n subregions, R_1, R_2, ..., R_n, such that:

1. $\bigcup_{i=1}^{n} R_i = R$.
2. R_i is a connected region, $i = 1, 2, ..., n$.
3. $R_i \bigcap R_j = \emptyset$ for all i and j, $i \neq j$.
4. $P(R_i) = TRUE$ for $i = 1, 2, ..., n$.
5. $P(R_i \bigcup R_j) = FALSE$ for any adjacent regions R_i and R_j

Here, $P(R_i)$ is the logical predicate that must be satisfied by the pixels in a segmented region. Thus, a pixel in the depth image with a corresponding grey value z_k is included in a region R_i if it fulfils:

$$|z_k - S| < T \tag{1}$$

where S is the grey-level value of the region 'seed' and T is a threshold that has been selected by empirical means. Once regions have been extracted from the background, they are described by means of the following features:

- Grey-scale image features. The number of pixels in the region is computed as an estimation of the region size. The region texture is defined by the entropy, H, expressing a low value a homogeneous distribution of the pixels. From the grey-level co-occurrence matrix, P, the entropy has been calculated as:

$$H = \sum_i \sum_j P(i, j) \cdot log_2[P(i, j)] \tag{2}$$

  where i and j are the number of rows and columns, respectively, in the co-occurrence matrix.
- Depth image features. A Radius-based Surface Descriptor (RSD) [18] is utilized as feature describing the geometry of the region. In the previous depth image processing, the oriented point cloud was computed and it is the input data for this descriptor. RSD estimates at each oriented point the maximum and minimum curvature radius taken from the distribution of normal angles by distance. The relation between the distance d of two points and the angle α between the point normals is:

$$d(\alpha) = \sqrt{2r} \cdot \sqrt{1 - cos\alpha} \tag{3}$$

  From this, it can be seen that given the distance and the angle between two point normals, the radius of the circle (r) can be approximated. In the case of a cylinder, this estimated radius will vary between the minimum radius (the radius of the cylinder) and the maximum radius (infinity). As it is intended to recognize cylindrical-shaped objects, the feature value of each point included in a region will be the minimum radius.

3.2.2 Feature fusion

Features of both images are normalized between 0 and 1 and included in a classifier to identify the object of interest. Hence, regions can be categorized into two classes: target (TG) and non-target (NTG). The TG class is identified as the object that should be tracked by the robot. Consequently, the NTG class is defined as other objects, such as the robot arm or hand, which are not considered as obstacles, and may be included in the 3D scene. A supervised learning classifier based on Artificial Neural Networks (ANN) has been used for classifying between classes. The classifier topology is as follows:

- Input layer: It is composed of neurons that present the two features extracted from the grey-scale image: area and texture, and the minimum radius for some of the oriented points included in the point cloud. For the depth features, the number of feature points included in this layer depends on the point cloud density.
- Output layer: This layer contains one neuron, and a value in its discriminant function next to one indicates that the assessed region belongs to the TG class, otherwise NTG class.
- Hidden layer: The number of nodes in the hidden layer has been approached by $Yei's$ formula [36]: $(a + b)/2$, where a is the number of input nodes and b is the number of output nodes.

The network training has been implemented using the back-propagation algorithm in its version with gradient descent, Gd. In addition, the activation function has been sigmoidal with value between [0,1]. Finally, two different criteria were applied to stop the training: in one case, it was stopped when the mean squared error (MSE) reached 0.01 and, in the other, after performing 1000 epochs. With the former criteria, the convergence of the algorithm is assured.

3.2.3 Object recognition

Examples of how the image processing proposal recognizes the 3D scene are shown in Fig. 3. The conveyor belt, considered as an obstacle, is identified and located in the depth image processing; however, the target recognition is not always feasible. When the target presents a clear cylindrical shape, it is easily recognized by RANSAC algorithm, even in the presence of partial occlusions caused by the robot arm movement (Fig. 3a). On occasions, RANSAC algorithm fails in target recognition. An example of this statement can be found in Fig. 3 b. Here, some blue points (the image processing results) are located in the target but others are on the robot hand. In this particular case, further image processing is required so fusing features from two different types of images improves the object

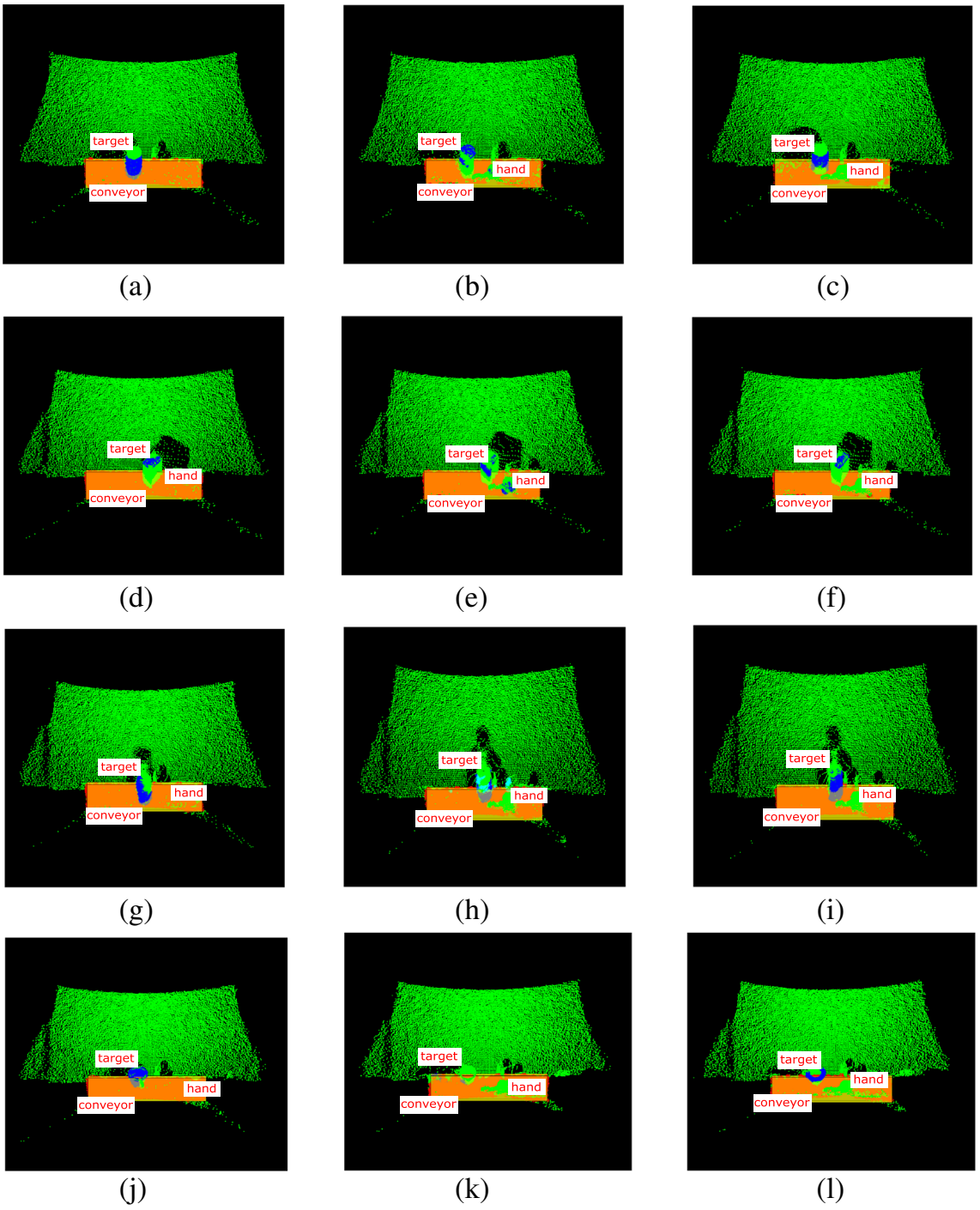


Fig. 3 Object recognition samples. **a**, **d**, **g**, **j** True detections in the depth image; **b**, **e**, **h**, **k** False detections in the depth image; **c**, **f**, **i**, **l** True detections with feature fusion

recognition. Now, the image processing algorithm with feature fusion recognizes the target (Fig. 3c, blue points are only on the target).

The image processing proposal has also been tested with objects not presenting a clear cylindrical shape. Figure 3 d shows the algorithm performance with a cube-shape target. Only some points are identified as target but when the robot arm is included in the 3D scene (Fig. 3e), the algorithm fails. Once feature fusion step is executed, the target is successfully located (Fig. 3f).

The algorithm has also been tested with everyday objects such as bottles and cups. Figure 3 g and j show true detections processing only the depth image. Once again, when the robot arm is included in the 3D scene, the

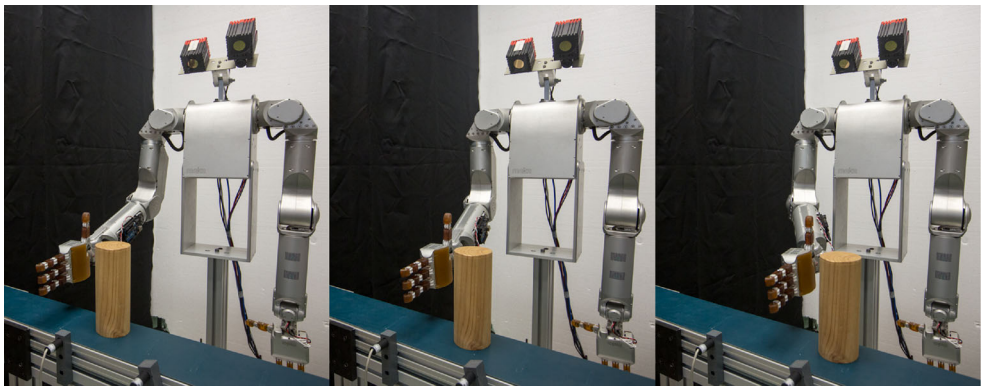


Fig. 4 Robotic platform for object tracking task

algorithm fails (Fig. 3h, k) and, finally, the target is properly located executing the feature fusion step (Fig. 3i, l).

4 Case study: object tracking task

An object tracking task has been implemented to validate the 3D object recognition proposal. As it is presented in this section, the proposal fulfils the two main performance requirements in robotic tasks: (1) high accuracy of detection and (2) an adequate processing speed. So, the robot task is to track a cylindrical-shaped object, the target, which is placed on a conveyor belt, utilized for its displacement.

The next subsections show the hardware set-up, software architecture and the tracking task detailing, first, the obstacle avoidance and, second, the robot arm positions while tracking the target.

4.1 Hardware set-up

Figure 4 shows the hardware set-up composed of a 7-d.o.f. Meka robot with a SR4000 ToF camera, placed on a static place at the top of the robot trunk. The SR4000 camera has a resolution of 176×144 pixels and covers a measurement range up to $10m$ with an absolute accuracy of $+/-15mm$ and a repeatability of $6mm$ depending on the working distance and the target reflectivity. Both, the Meka robot and the conveyor belt are fixed on a solid surface.

Through the image processing, the target and obstacles are recognized but in the camera frame. To estimate position and orientation in the robot frame, the ToF camera has to be calibrated. Subsequently, this information is used to control the robot arm movement.

4.2 Software architecture

In reference to the software architecture, it is running over ROS (Robot Operating System) platform [37], which is the most widespread robotics-specific communication middleware platforms. ROS although not being a component-based platform, it is formed by modular programming units which are called nodes. In order to provide external accessibility, there is the topic concept. A node interested in making information accessible publishes this data via topics. When another node is interested in a certain kind of data, it will subscribe to the appropriate topic.

Figure 5, by an UML component diagram, details the main nodes (UML components in the figure) running over ROS information that gives accessibility to some topics, but also, in some cases, they subscribe to others. Hence, *SR4000* is responsible for providing both grey and depth images. The *Bounding Box* node requires the depth image and performs previously commented algorithmic (see Fig. 1) to obtain and provide the position of the conveyor belt and the simplified and oriented point cloud. The *Target_Pos* node requires the simplified point cloud and the grey-scale image provided by *BoundingBox* node and *SR4000*, respectively, to obtain and publish the target position.

Meka follows a trajectory generated by an MPC (Model Predictive Controller)-based trajectory planner algorithm [10]. To generate this trajectory, *MPC_Controller* node requires the current position of Meka robot, the constraints positions and the position of the target (which is in movement). It also manages the transfer function that models the behaviour of the Meka Robot.

Finally, Fig. 6 details the software deployment in the distributed platform, which is formed by two main PCs. In PC1, it is running the code responsible for obtaining the trajectory to be followed by Meka. The code responsible for MEKA movement is running in PC2. The PCs information exchange is solved by ROS. The authors want to note that for software modelling, the ART2ool has been used [38].

4.3 Object tracking

The result of the object tracking task with collision avoidance is shown in Fig. 7. First, a detail of how the robot arm adapts its trajectory avoiding the collision with the conveyor belt is presented in Fig. 7 a. Within the initial trajectory, the robot arm enters in the forbidden 3D

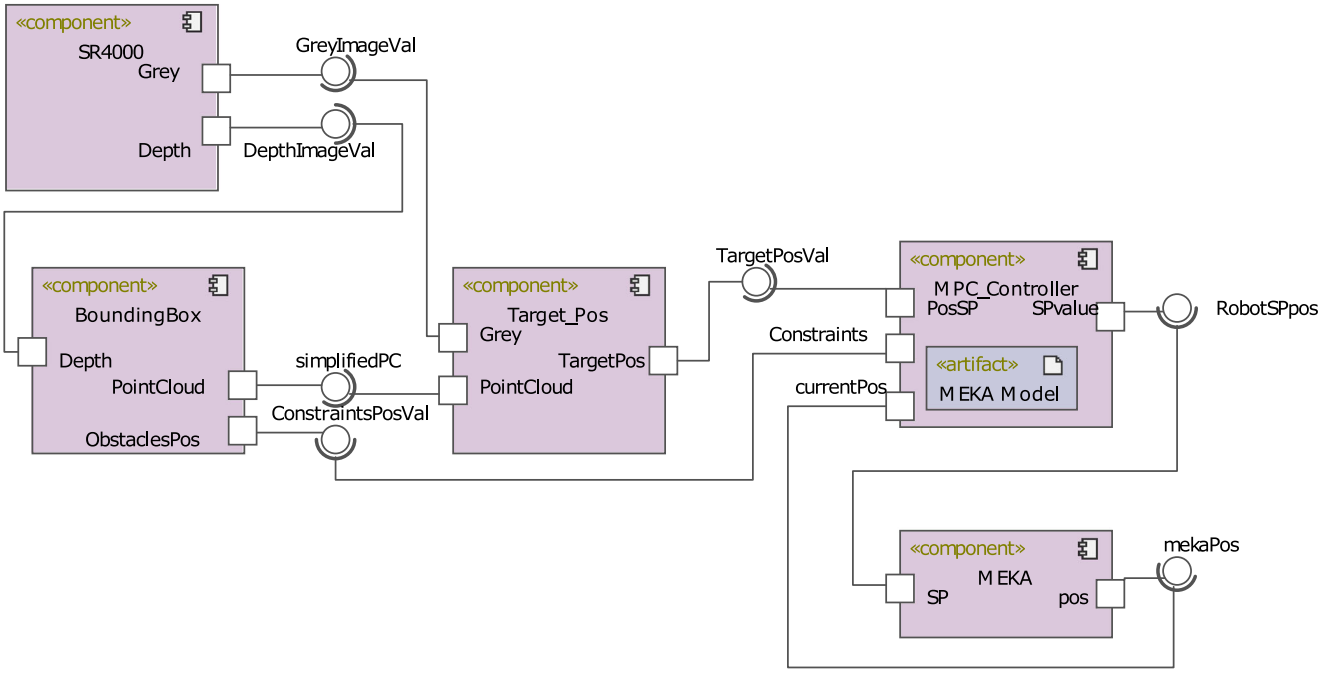


Fig. 5 Software ROS nodes and their data interchange by UML component diagram

region, the conveyor belt bounding box. Considering this information, the MPC controller adapts the robot movement generating a new reference trajectory. The result is that the robot trajectory follows this new reference avoiding any collision.

Second, Fig. 7 b shows a detail of the robot arm movement preforming the tracking task. Here, black dots correspond to the target positions, identified through the ToF camera and translated its coordinates into the robot frame. Note that the target was properly located in the whole camera field of view and the robot hand—green trajectory—followed the target displacement. As the robot started its movement from a home position, any kind of collision with the conveyor belt was avoided. Therefore, the green trajectory was not included in the red bounding box that represents the conveyor belt, even if the reference—blue

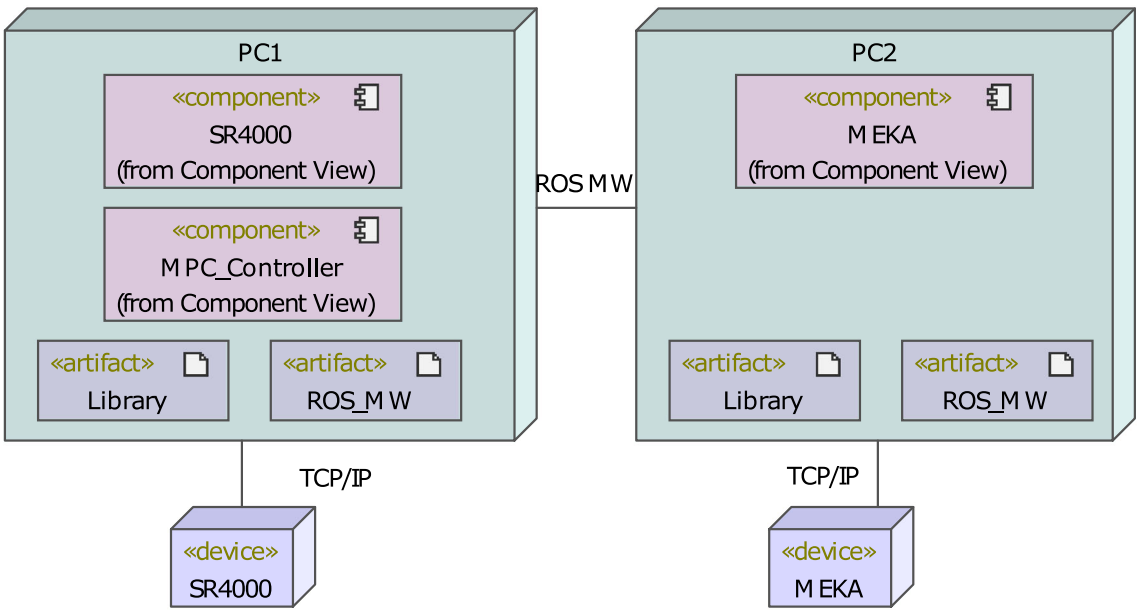
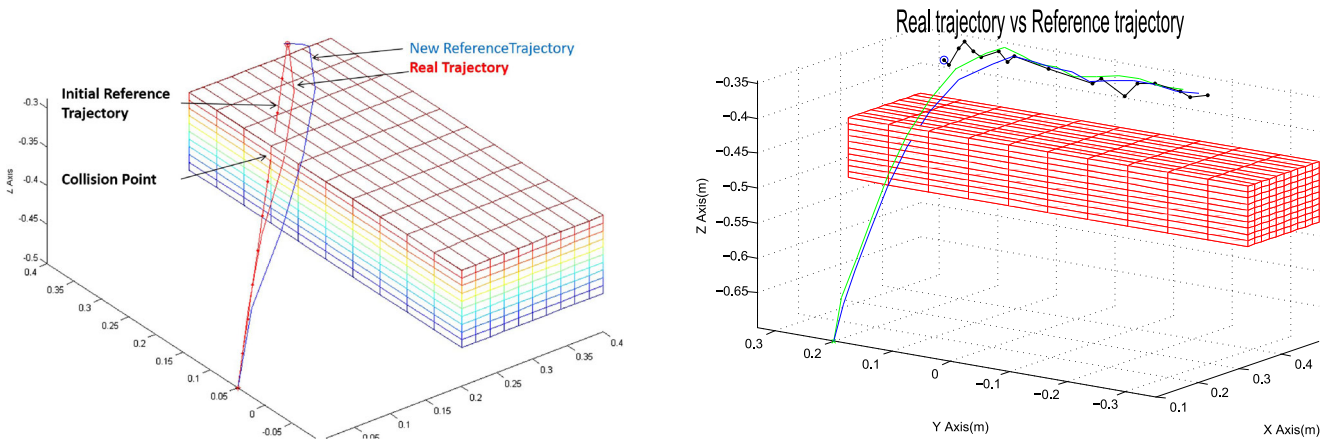


Fig. 6 Software deployment in the distributed platform by UML deployment diagram



(a) (b)

Fig. 7 Object tracking with collision avoidance. **a** Collision avoidance; **b** Object tracking

trajectory—intersected the conveyor belt bounding box. With these tests, it is clearly shown how the constraints, included in the visual control strategy, reject any kind of collision during the arm movement.

5 Conclusion

We have presented a 3D object recognition algorithm based on feature fusion, specially designed for robot vision tasks. A time-of-flight camera was utilized as an image acquisition device. ToF cameras provide depth and grey-scale information at a high frame rate, and this is what makes them suitable for real-time applications.

To deal with image processing, a two-layer classification scheme which identifies and locates cylindrical-shaped objects was developed. Firstly, information was extracted from depth images and objects, not included into the robot workspace, and potential obstacles were recognized and then removed from the input data. Removing unnecessary information reduces computation time and simplifies further processing. Subsequently, an attempt to recognize the target was made using the RANSAC algorithm for cylinders. On occasions, and mainly depending on the sensor point of view, this algorithm failed and new image processing strategies were required.

So, secondly, the grey-scale image was analysed and low-level information, such as object intensity or texture, was obtained. These features along with a rough description of the object geometry were utilized for its recognition. As can be seen from image processing results, fusion of information gathered from different sources increased the effectiveness of the recognition.

The proposal was validated on a robotic platform which consisted on an anthropomorphic 7-d.o.f. Meka robot, a SR4000 ToF, and a conveyor belt. A challenging problem of robot vision is an object tracking task that was chosen for testing the developed computer vision algorithms.

Funding information This work has been partially supported by the project DPI2016-78290-R.

References

1. Robla-Gómez S, Becerra VM, Llata JR, González-Sarabia E, Torre-Ferrero C, P3́rez-oria J (2017) Working together: a review on safe human-robot collaboration in industrial environments. IEEE Access 5:26754–26773
2. Alzarok H, Fletcher S, Longstaff A (2017) 3D Visual tracking of an articulated robot in precision automated tasks. Sensors 17(104):1–23
3. Hoiem D, Savarese S (2011) Representations and techniques for 3D object recognition and scene interpretation. Morgan & Claypool, San Rafael
4. Mian AS, Bennamoun M, Owens R (2006) Three-dimensional model-based object recognition and segmentation in cluttered scenes. IEEE Trans Pattern Anal Mach Intell 28(10):1584–1601
5. Foix S, Aleny G, Torras C (2011) Lock-in time-of-flight (ToF) cameras: a survey. IEEE Sensors J 11(9):1917–1926
6. Jain AK, Dorai C (2000) 3D object recognition: representation and matching. Stat Comput 10:167–182
7. Guo Y, Bennamoun M, Sohel F, Lu M, Wan J (2014) 3D Object recognition in cluttered scenes with local surface features: a survey. IEEE Trans Pattern Anal Mach Intell 36(11):2270–2287
8. Hussmann S, Liepert T (2009) Three-dimensional TOF robot vision system. IEEE Trans Instrum Meas 58(1):141–146
9. Forsyth DA, Ponce J (2003) Computer Vision: a modern approach. Prentice Hall, Upper Saddle River
10. Satorres Martínez S, Gómez Ortega J, Gámez García J, Sánchez García A, De la Casa cárdenas J (2013) Visual predictive control of robot manipulators using a 3D ToF camera. In: IEEE International Conference on Systems, Man, and Cybernetics, pp 3657–3662
11. Satorres Martínez S, De la Casa cárdenas J, Gámez García J, Gómez Ortega J (2014) Position predictive control of an

anthropomorphic robotic arm using a time-of-flight camera. IEEE Sensors pp 1718–1721
12. Andreopoulos A, Tsotsos JK (2013) 50 years of object recognition: directions forward. Comput Vis Image Underst 117:827–891
13. Rusu RB, Blodow N, Beetz M (2009) Fast Point Feature Histograms (FPFH) for 3D registration. In: Proceedings of the International Conference on Robotics and Automation (ICRA)
14. Rusu RB, Marton ZC, Blodow N, Beetz M (2008) Learning informative point classes for acquisition of object model maps. Proceedings of the 10th International Conference on Control, Automation, Robotics and Vision (ICARCV), pp 17–20
15. Rusu RB, Marton ZC, Blodow N, Dolha M, Beetz M (2008) Towards 3D point cloud based object maps for household environments. Robotics and Autonomous System Journal (Special Issue on Semantic Knowledge)
16. Rusu RB, Holzbach A, Beetz M (2009) Detecting and segmenting objects for mobile manipulation. In: International Conference on Computer Vision Workshops, pp 47–54
17. Rusu RB, Bradski G, Thibaux R, Hsu J (2010) Fast 3D recognition and pose using the Viewpoint Feature Histogram. In: International Conference on Intelligent Robots and Systems (IROS), pp 2155–2162
18. Marton Z-C, Pangercic D, Blodow N, Kleinehellefort J, Beetz M (2010) General 3D modelling of novel objects from a single view. In: International Conference on Intelligent Robots and Systems (IROS), pp 3700–3705
19. Johnson AE, Hebert M (1999) Using spin images for efficient object recognition in cluttered 3D scenes. IEEE Trans Pattern Anal Mach Intell 21(5):433–449
20. Frome A, Huber D, Kolluri R, Bülow T, Malik J (2004) Recognizing objects in range data using regional points descriptors. ECCV 3:224–237
21. Tombari F, Salti S, Di Stefano L (2010) Unique shape context for 3D data description. In: Proceedings on the ACM workshop on 3D object retrieval, pp 57–62
22. Arbeiter G, Fuchs S, Bormann R, Fischer J, Verl A (2012) Evaluation of 3D feature descriptors for classification of surface geometries in points clouds. In: International Conference on Intelligent Robots and Systems (IROS), pp 1644–1650
23. Lazebnik S, Schmid C, Ponce J (2005) A Sparse texture representation using local affine regions. IEEE Trans Pattern Anal Mach Intell 27(8):1265–1278
24. Hussmann S, Liepert T (2009) Three-dimensional TOF robot vision system. IEEE Trans Instrum Meas 58(1):141–146
25. Tang K, Song P, Chen X (2017) 3D object recognition in cluttered scenes with robust shape description and correspondence selection. IEEE access, Special Section on Trends and Advances for Ambient Intelligence with Internet Of Things (IOT) Systems 5:1833–1845
26. Li Y, Wang S, Tian Q, Ding X (2015) Feature representation for statistical-learning-based object detection: a review. Pattern Recogn 48:3542–3559
27. Castellini C, Tommasi T, Noceti N, Odone F, Caputo B (2011) Using object affordances to improve object recognition. IEEE Trans Auton Ment Dev 3(3):207–215
28. Homberg BS, Katzschmann RK, Dogar MR (2015) Haptic identification of objects using a modular soft robotic gripper. In: International Conference on Intelligent Robots and Systems, IEEE, pp 1698–1705
29. Andreopoulos A, Tsotsos JK (2013) A computational learning theory of active object recognition under uncertainty. Int J Comput Vis 101:95–142
30. Saidi F, Stasse O, Yokoi K, Kanehiro F (2007) Online object search with a humanoid robot. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp 1677–1682
31. Sujan VA, Dubowsky S (2005) Efficient information-based visual robotic mapping in unstructured environments. Int J Robot Res 24(4):275–293
32. Ray PP (2016) Internet Of robotic things: concept, technologies, and challenges. IEEE Access 4:9489–9500
33. Pundlik S, Patii V, Navalgund M, Sahasrabudhe A, Shinde S (2018) Object tracking bot using on-board Jetson TK1: an approach to reduce communication overhead and time delay. In: International Conference on Big Data, IoT and Data Science. https://doi.org/10.1109/BID.2017.8336575
34. Schnabel R, Wahl R, Klein R (2007) Efficient RANSAC for point-cloud shape detection. Comput Graphics Forum 26(2):214–226
35. Russ JC (2007) The image processing handbook. CRC Press Taylor & Francis Group, Chapman & Hall
36. Duda RO, Hart PE, Stork DG (2001) Pattern classification. Wiley, New York
37. Russell J, Cohn R (2012) ROS (robotic operating system). VSD
38. Estévez E, Sánchez García A, Gámez García J, Gómez Ortega J (2018) 'ART2ool: a model-driven framework to generate target code for robot handling tasks. Int J Adv Manuf Technol 97(1):1195–1207