Unsupervised Feature Extraction from RGB-D Data for Object Classification: a Case Study on the YCB Object and Model Set

André Brás
Dept. of Mechanical Engineering, University of Coimbra
Coimbra, Portugal
Email: andre.bras@uc.pt

Pedro Neto
Dept. of Mechanical Engineering, University of Coimbra
Coimbra, Portugal
Email: pedro.neto@dem.uc.pt

***Abstract*—Object recognition has attracted increasing attention of researchers due to its numerous applications. For instance, it enables robots to carry out tasks like searching for an object in an unstructured environment or retrieving a tool for a human co-worker. In this study, we present a new technique for unsupervised feature extraction from red, green, blue, plus depth (RGB-D) data, which is then combined with several classifiers to perform object recognition. Specifically, our architecture segments all objects in a table top scene through an unsupervised clustering technique. It focuses separately on each object to extract both shape and visual features. We conduct experiments on a subset of 20 objects selected from the YCB object and model set and evaluate the performance of several classifiers. The most effective one achieves an accuracy of 99.7% when trained and tested on samples acquired with the same conditions (equipment and environment). Results degrade when the system is trained with YCB data and tested with data acquired from a Kinect sensor in online laboratorial implementation.**

***Index Terms*—object recognition; shape features; RGB-D; lightweight algorithm; robotics**

I. Introduction

As the presence of collaborative robots in highly dynamic workspaces becomes more prevalent, autonomy and advanced cognitive reasoning are increasingly a demand for these robots. To achieve this end, the robots will need to be aware of many physical attributes of the world, just as the humans do [18]. For instance, humans see novel objects and are instantly capable of recognizing them. In other words, they segment a scene into parts, describe what those parts are, and distinguish visually similar objects in real time. Similarly, recognizing object instances and categories is also a crucial ability for autonomous and collaborative robots to understand and interact with the physical world. Object recognition and detection has thus been a major focus of research in computer vision, machine learning and robotics communities [6], [9].

Even though object recognition is a fundamental problem for several scientific communities, it remains challenging since objects can exhibit large variations in appearance due to changing viewpoints, deformations, scales, and lighting conditions. Hence, numerous features and approaches have been developed and applied to this problem, contributing to important breakthroughs in the state-of-the-art.

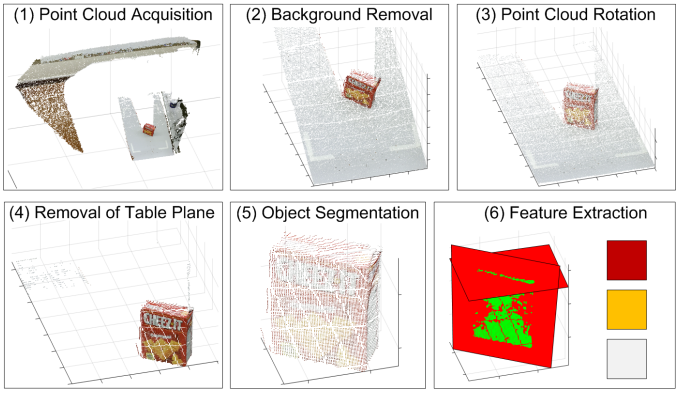

Fig. 1: Overview of the proposed approach.

Over the last years, a new generation of sensing technologies capable of providing RGB-D images have emerged. Compared to red, green, and blue (RGB) data, which provide objects' appearance information, depth data contain additional cues about its geometric shape. Besides, depth data are invariant to lighting and color variations. Nowadays, RGB-D sensors are relatively inexpensive, having favored the availability of large image datasets [5], [9], [19] and the development of features for color and depth channels. Hence, they represent an opportunity to improve RGB-D data-based object recognition, which is at the core of many robotic applications. Such ability enables robots to make informed decisions in achieving tasks like retrieving an object from a novel environment for a human.

This study introduces a framework for unsupervised feature extraction from RGB-D data. A major objective is related to the ability to acquire RGB-D features that serve as input to traditional classifiers and which compete with deep learning in accuracy, training time and not requiring large datasets. Such ability may facilitate the online implementation of object recognition in the most diverse domains. Our approach segments objects previously placed above a table and extracts both shape and visual features from each one. Specifically, the method fits primitive geometric shapes (cylinder, sphere,

and rectangular prism) to each object and uses the best model to estimate its volume. Besides, k-means clustering draws upon RGB data to reveal main colors of the objects. These representations and the corresponding object class are used as inputs to different classifiers. We validate our approach on the Yale-Carnegie Mellon University (CMU)-Berkeley (YCB) object and model set [5]. The key contributions of this study can thus be summarized as follows:

- Object segmentation and feature extraction which can accurately detect and recognize several objects simultaneously;
- Robust shape and visual features able to deal with unstructured environments and lack of knowledge about background and object pose;
- We evaluate our approach on a subset of 20 objects selected from the YCB object and model set;
- Source code is available online at https://github.com/AndreBrasUC/Object_Recognition_From_RGBD_Data.

II. Related Work

Since our work is related to a large body of studies on feature extraction and object recognition from RGB-D data, we will highlight a few connections and differences between our approach and the most recent literature. Early approaches for object recognition in three-dimensional (3D) environments focused on developing handcrafted features from the point cloud space or the associated RGB-D data. On the one hand, the 3D point cloud representation allowed the extraction of local features, such as the Signature of Histogram of Orientations [20], as well as global features, such as the Viewpoint Feature Histogram [17]. On the other hand, many feature descriptors have been applied directly on RGB-D data, namely the Scale Invariant Feature Transform (SIFT) [9], [13], spin images [7], [9], Speeded Up Robust Features (SURF) [2], Histograms of Oriented Gradients (HOG), and Bag-of-Words (BoW). Lai et al. [9] presented the RGB-D Object Dataset and an object recognition and detection framework. Their object recognition approach used handcrafted descriptors, such as spin images [7] and SIFT [13], to retrieve shape and visual representations, respectively. Afterwards, they evaluated the performance of 3 classifiers considering shape features, visual features and both shape and visual features. They concluded that combining color and depth information improves the results. In our study, we also split shape and visual features to evidence the significance of each subset and the benefits of using them together. Lai et al. [10] applied the same set of shape and visual features to propose and evaluate the Instance Distance Learning (IDL) classifier. Such technique resembles the nearest neighbor approach, which labels an incoming test image using the label of the nearest instance, but it exhibits two key differences. Firstly, each novel view of an object is simultaneously compared to all views of a previous object, resulting in significant computational cost savings. Secondly, IDL classifier replaces a regular distance function between views by a learned instance distance function, which allows the classifier to assign different weights to each feature and for each view.

A few years ago, several research groups started proposing unsupervised feature learning algorithms to RGB-D data-based object recognition, opening a new perspective of feature extraction techniques [2], [3]. Blum et al. [2] proposed the convolutional k-means descriptor, which uses RGB-D images and learns local features from the neighborhood of interest points previously detected by the SURF descriptor. Our study also uses k-means descriptor, yet with a few different details. Specifically, we apply k-means descriptor on RGB data to get the main colors of each object. Bo et al. [3] introduced the hierarchical matching pursuit (HMP) method, which can build features in an unsupervised fashion from RGB-D images.

Even though unsupervised feature learning techniques have shown good performance on object recognition problems, they are still based on handcrafted features. Such approaches conceal at least two basic limitations. On the one hand, during the learning process, these features usually only capture a small set of recognition cues from raw data, ignoring other equally useful information. On the other hand, handcrafted features must be redesigned for new data types, making object recognition systems heavily dependent on expert experience [3]. Therefore, the emergence of effective and efficient learning algorithms able to learn robust representations from raw data was a natural evolution. Accordingly, deep learning methods are representation learning methods which allow a machine to be fed with raw data and automatically discover representations needed for detection or classification tasks [12]. The learned features have shown valuable results in a myriad of domains, being many times better than those achieved with engineered descriptors [1].

Convolutional Neural Networks (CNNs) [11] are examples of deep networks used to learn latent and complex features directly from data. The usage of CNNs in [8] to almost halve the error rate for object recognition was a breakthrough that triggered the application of deep learning by multiple communities to other object recognition problems [4], [6], [14]–[16]. Redmon and Angelova [16] adapted the widely used CNN proposed in [8] and introduced an approach for simultaneous object recognition and robotic grasp detection from RGB-D images. Motivated by the advances of machine learning and computer vision communities, Eitel et al. [6] proposed a novel RGB-D architecture for object recognition. Specifically, they applied 2 CNNs operating separately on color and depth modalities, being then combined by a late fusion approach.

While applying pre-trained networks for object recognition from RGB data is straightforward, using such networks for processing depth data is not. Therefore, researchers have proposed depth data encoding methods to enable the reuse of CNNs previously trained on RGB data. Most of the multi-stream approaches have combined RGB data and a single depth data encoding method. However, Rahman et al. [15] presented a novel multimodal architecture for object recognition which encodes depth data with 3D surface normals and

jet colormap, splitting thus depth data into 2 streams. Then, they combined both depth streams with RGB data to build a deep network composed of 3 branches.

An unprecedented and attractive approach for object recognition was proposed in [4]. The authors developed a region proposal method able to locate objects in a 3D point cloud representation through an unsupervised clustering technique. From the clusters, they computed the 3D position of each object and derived the coordinates of a bounding box, which was then translated into the corresponding position in RGB space. Finally, the defined patches were fed into a CNN for classification. In our work, we also use an unsupervised clustering technique. However, they segmented objects using a Euclidean clustering algorithm, while we consider the total number of points and its density in each cluster. Furthermore, we learn features directly from these cluster and use them in a feedforward ANN, while they use a CNN with RGB data.

III. Unsupervised Feature Extraction

We introduce a novel unsupervised feature extraction algorithm which uses RGB-D data to distinguish between objects. According to Fig. 1, we use a Microsoft Kinect sensor to acquire an RGB-D image of the table top scene and the corresponding point cloud. These data are used for testing (online implementation), since the training data is obtained from the YCB dataset. In this study, we focus only on the object recognition problem for table top settings as they are common environments. Afterwards, we take advantage of the previous knowledge about the table position to trim the background and obtain a point cloud composed only by the table and the objects. The point cloud is then rotated so that the table is parallel to a cartesian plane, the XOZ plane, which eases the removal of table points. Object segmentation is completed by isolating each object (cluster) in its own point cloud. Then, for each object, we examine its geometry and color to extract shape and visual features, respectively.

A. Point Cloud Processing and Object Segmentation

The proposed segmentation method, Algorithm 1, outputs a set of clusters in which each one confines an object in the table top scene. These clusters serve as inputs to feature extraction.

The inputs to the algorithm are a point cloud, $\mathbf{C}$, and a bounding box, $\mathbf{b}$. A point cloud is a m-by-n-by-3 matrix, where m and n are defined by the Microsoft Kinect's RGB sensor resolution. The bounding box is the mathematical expression of the knowledge about the target 3D space. So, $\mathbf{b}$ is a row vector which includes the center position and the dimensions of a rectangular prism. Even though Microsoft Kinect is a good RGB-D camera, it is still affected by noise and missing depth data, mainly due to reflective properties of materials. Therefore, we remove invalid points (Line 2). After such operation, the point cloud becomes a p-by-3 matrix, where p is the total number of valid points. In this moment, we downsample the point cloud to ensure real-time processing and sufficient coverage (Line 3). A proper sampling parameter, α, which represents the remaining portion of point cloud, is equal to 0.60. Subsequently, points' position is evaluated and only those belonging to the rectangular prism $\mathbf{b}$ are stored, which allows us to reach a much smaller point cloud, $\mathbf{D}$ (Line 4). This point cloud is composed only by the table and the objects. Since Microsoft Kinect sensor is not aligned with the horizontal plane, the table shows up in the point cloud in a diagonal position. However, as our setting is static, this angle can be roughly measured. The further step is thus rotating the point cloud $\mathbf{D}$ to obtain another point cloud, $\mathbf{E}$, where the table plane is almost aligned with XOZ plane (Line 5). Accordingly, $\mathbf{R}$ is a rotation matrix describing a rotation of 30º around the x axis. Then, we use M-estimator SAmple Consensus (MSAC), which is a variant of the RANdom SAmple Consensus (RANSAC) algorithm, to fit a plane to the table. $\mathbf{T}_{inliers}$ includes all points whose distance to the computed plane is lower than a predefined value. The y axis is then aligned with the normal vector, $\mathbf{n}$, of the plane and we get a new point cloud, $\mathbf{F}$, where the table is finally parallel to XOZ plane (Lines 6-8). Afterwards, we plot a histogram which counts the number of points along y axis. Since the table is the biggest provider of points, we can find the maximum value, add an offset, and remove all table points (Line 9). The point cloud $\mathbf{G}$ contains only the objects and a few outliers, which we eliminate through a pair of equal denoise operations (Lines 10-11). A given point is an outlier if the average distance to its β-nearest neighbors is above a specified threshold. Decreasing the number of nearest neighbors makes the filter more sensitive to noise and relevant data can be removed too. Increasing this value increases the number of computations. We are considering the 75 nearest neighbors as this value has given positive results. Furthermore, the threshold is one standard deviation above the mean of the average distance to β-nearest neighbors of all points. Finally, we split XOZ plane in small square cells, count the points in each cell, and look for non-empty adjacent cells, which represent a segmented object (Line 12).

Algorithm 1 Point Cloud Processing and Object Segmentation

1: **Given** point cloud $\mathbf{C}$, bounding box $\mathbf{b}$
2: $\mathbf{C} \leftarrow$ remove invalid points ($\mathbf{C}$)
3: $\mathbf{C} \leftarrow$ downsample ($\mathbf{C}$, α)
4: $\mathbf{D} \leftarrow \mathbf{C} \cap \mathbf{b}$
5: $\mathbf{E} \leftarrow \mathbf{R}\ \mathbf{D}$
6: $\mathbf{T}_{inliers} \leftarrow$ MSAC ($\mathbf{E}$)
7: $\mathbf{n} \leftarrow$ get normal vector ($\mathbf{T}_{inliers}$)
8: $\mathbf{F} \leftarrow$ rotate $\mathbf{E}$ so that y axis is aligned with $\mathbf{n}$
9: $\mathbf{G} \leftarrow$ remove table plane ($\mathbf{F}$)
10: $\mathbf{G} \leftarrow$ denoise ($\mathbf{G}$, β)
11: $\mathbf{G} \leftarrow$ denoise ($\mathbf{G}$, β)
12: $\mathbb{O} \leftarrow$ cluster ($\mathbf{G}$)
13: **return** $\mathbb{O}$

B. Feature Extraction (shape and visual features)

Feature extraction (shape and visual features) is individually applied to each object previously segmented.

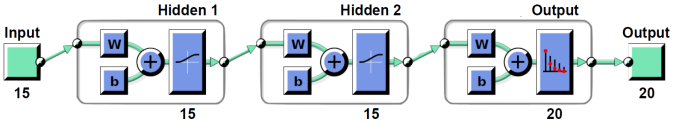

Fig. 2: Network architecture used for object classification. The input layer has 15 units, as well as both hidden layers. The output layer has 20 units.

We use MSAC algorithm to fit cylinders and spheres to each object. Then, we compare the number of inliers and the total number of points to evaluate the quality of each model. A given point is an inlier if the distance to the model is lower than a specified threshold. Furthermore, we also adjust a rectangular prism to the point cloud. Since MSAC algorithm can not do it directly, we first fit the frontal and top planes. If the number of inliers of these planes is considerable when faced to the total number of points, we finish the prism by modelling the other 4 planes. After we have adjusted these 3 primitive geometric shapes, we use the model with highest quality to compute object's volume. Therefore, we compute 3 shape features, namely (1) the primitive geometric shape that best fits the object, (2) the corresponding quality, and (3) the object's volume. However, when all models are poor, we assume that no geometric shape properly describes the model and point cloud's volume is used instead of object's volume.

Since RGB-D data is available (from the YCB and Microsoft Kinect), we can build point clouds composed of colored points and use all values of RGB channels to obtain visual features of objects in the table top scene. On one hand, we apply k-means clustering with k equal to 1 to obtain the centroid of all points in RGB space. This centroid is represented by 3 values, 1 for each channel, and expresses the main color of the point cloud. On the other hand, we use k-means clustering with k equal to 3 to retrieve 3 predominant colors. Altogether, we extract 12 visual features (1 main color + 3 secondary colors) from each object.

C. Classification

Feature vectors are taken as input by a feedforward ANN, whose architecture is shown in Fig. 2. Each feature vector includes 3 shape features and 12 visual features, and thus the input layer has 15 units. The network comprises 2 hidden layers and both contain 15 units. Furthermore, both layers apply sigmoid as activation function. The output layer has 20 units, which is the number of classes, and uses the softmax activation function. Finally, this network is optimized with the Scaled Conjugate Gradient (SCG). The other classifiers are a decision tree with 100 leaf nodes, linear, quadratic, and cubic Support Vector Machines (SVM), and also 2 different k-nearest neighbors (k-NN) algorithms.

IV. EXPERIMENTS AND RESULTS

A. YCB object and model set

To illustrate the effectiveness of our approach, we conduct our experiments on the YCB object and model set [5]. Such

Fig. 3: Subset of 20 objects selected from the YCB object and model set. From back to front and left to right: cracker box, chips can, wood block, bleach cleanser, pitcher, potted meat can, master chef can, sugar box, mustard bottle, bowl, gelatin box, tomato soup can, tennis ball, softball, baseball, plum, peach, apple, orange, and lemon.

dataset has been made available to facilitate benchmarking in robotic manipulation research. YCB object and model set includes 77 different objects divided into 5 categories, namely food items, kitchen items, tool items, shape items, and task items. From the dataset we selected a subset of 20 objects, Fig. 3, based on a survey of the most common items used in literature considering a variety of shapes, sizes, textures, weights, rigidities, transparencies, as well as the difficulty of grasping and manipulating them. The associated database provides RGB-D images, physical properties, and geometric models of the objects. Additionally, the scoring schemes for the quantification of performance are also available. These protocols enable the community to compare approaches and evolve standardized benchmarking tests and metrics.

Visual data were collected with the scanning rig used to build BigBIRD dataset [19]. The rig encompasses 5 RGB-D sensors and 5 high-resolution RGB cameras arranged in a quarter-circular arc. Each object is placed on a computer-controlled turntable, which is rotated by 3º at a time, yielding 120 turntable orientations. Therefore, the setup yields 600 RGB-D images and 600 high-quality RGB images for each object, being each one followed by the corresponding segmentation mask. However, we only use data from the second and third cameras, when counting from the horizontal, which gives us 240 RGB-D images and the same number of high-quality RGB images of each object.

B. Evaluation

In SCENARIO 1 the system is trained and tested with YCB data. In SCENARIO 2 the system is trained with YCB data and tested with data acquired from a Kinect sensor in online laboratorial implementation. Fig. 4 shows an example of the extracted shape features for the orange and visual features for the cracker box. In Table I, we present the classification results

TABLE I: Accuracy from 7 different classifiers considering shape features, visual features, and shape + visual features.

Classifier	Shape Features	Visual Features	Shape+visual Features
Proposed ANN	56.1	98.5	99.7
Decision Tree	65.0	92.3	96.4
Linear SVM	43.6	96.5	98.2
Quadratic SVM	48.5	99.5	99.3
Cubic SVM	19.1	99.4	99.1
1-NN	59.7	99.3	98.5
5-NN	64.0	97.8	96.6

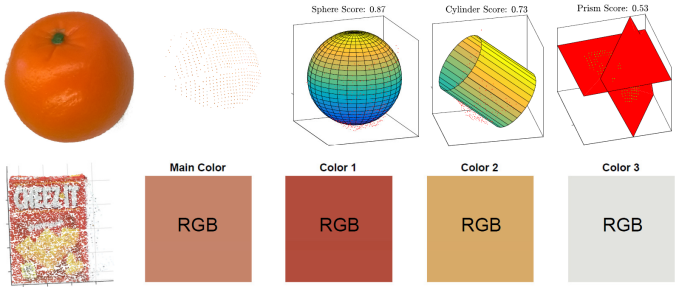

Fig. 4: Example of shape and visual features captured for 2 different objects (orange and cracker box).

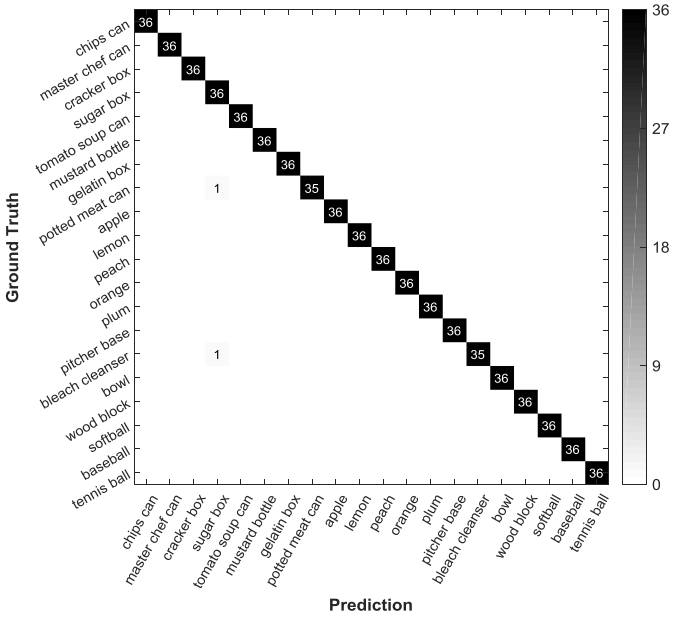

Fig. 5: Confusion matrix achieved by the feedforward ANN for 720 RGB-D images provided along with the dataset (SCENARIO 1).

achieved having different features as input. When we trained the ANN, we randomly split the total number of samples of each object class into training (70%), validation (15%), and test (15%) samples. All the other classifiers were trained with a 5-fold cross validation technique to prevent overfitting.

We trained several architectures for the feedforward ANN, namely networks with different number of hidden layers and different number of units in these layers. We realized that networks with 2 hidden layers behave slightly better than networks with a single hidden layer. At the end, the strategy detailed in Section III-C achieves a maximum accuracy of 99.7% on test samples. We used the same architecture with only shape features and only visual features, which enabled us to achieve accuracies of 56.1% and 98.5%, respectively. So, we prove that more than a half of the total test samples can be well classified with only 3 shape features. This is a promising result as several objects have similar shape and size, such as the baseball, the tennis ball, and all the plastic fruits. Furthermore, visual features alone have shown positive performance, even considering objects with similar colors, such as the chips can and the cracker box, or the baseball and the softball.

Fig. 5 shows the confusion matrix outputted by the feedforward ANN when we used it to perform object recognition on the 720 test samples with both shape and visual features (SCENARIO 1). This figure reveals only 2 misclassifications. Firstly, the algorithm predicted the sugar box instead of the potted meat can, which can be explained by the presence of the yellow and blue colors on both objects. Secondly, the bleach cleanser was also confounded with the sugar box because white, yellow and blue are common colors.

After we have trained and evaluated all the classifiers on visual data provided with the dataset, we acquired 10 RGB-D images of each object in Fig. 3 with a Kinect sensor (SCENARIO 2). Then, we used the previously trained feedforward ANN to predict a label for these 200 RGB-D images. We achieved an accuracy of 34.0% and the corresponding confusion matrix is in Fig. 6. When we used only shape features and only visual features, we achieved accuracies of 19.5% and 22.0%, respectively. Even though these results are divergent from those presented above, there is a simple explanation. Visual data provided with the dataset were not acquired with a Kinect sensor and we can confirm that visual features extracted from the available visual data are very different from those computed from the RGB images acquired with the Kinect sensor. This fact explains, for instance, the confusion between the plum (purple) and the apple (red), or between the lemon (yellow) and the tennis ball (green). It is expected that the algorithm works well when trained and evaluated on samples acquired with the same equipment.

The experiments described above were carried out with a single object placed on the table. However, the proposed framework can deal with multiple objects. Accordingly, in Fig. 7, we show 4 well-segmented objects from a single point cloud.

V. Conclusion and Future Work

In this study we presented and demonstrated a simple, and reliable object recognition framework. It can detect objects in an unstructured environment, extract shape and visual features from each object, and output the corresponding label. We proved that this technique can achieve an accuracy of 99.7% when it is trained and tested on samples acquired with the same camera. The system demonstrated reliability to segment

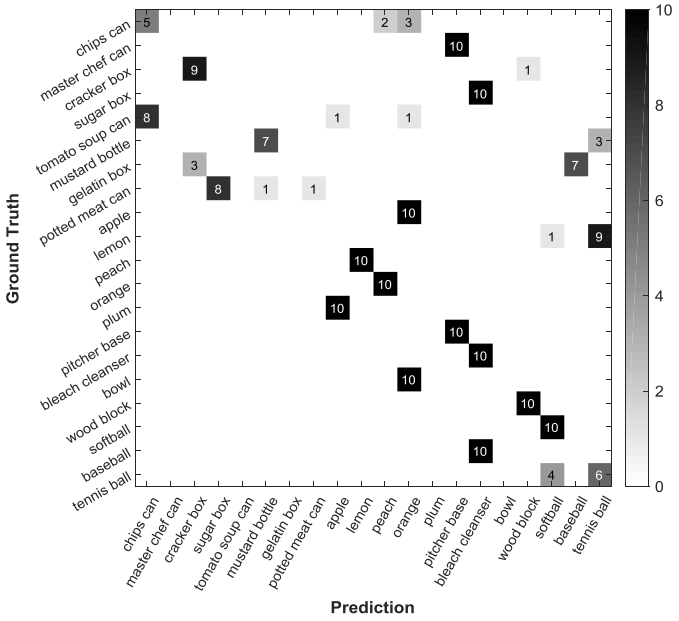

Fig. 6: Confusion matrix achieved by the feedforward ANN for 200 RGB-D images acquired with the Kinect sensor (SCENARIO 2).

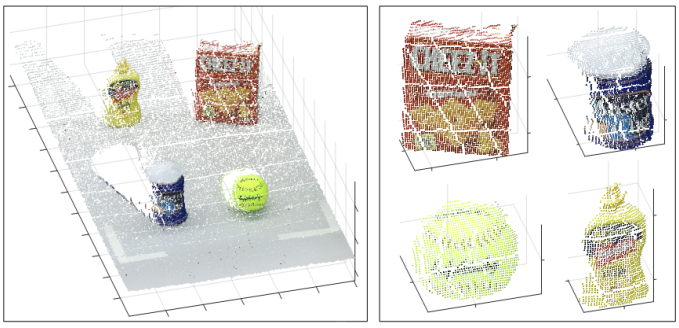

Fig. 7: Object segmentation. The point cloud on the left side shows the table and the objects after rotation. On the right side, all objects were already segmented and they can serve as inputs to feature extraction.

different objects in given scene. In future work, we will expand our results by providing a comparative analysis with deep learning.

ACKNOWLEDGMENTS

This work was supported in part by European Union's Horizon 2020 research and innovation programme under grant agreement No 688807 (ColRobot project), the Portuguese Foundation for Science and Technology (FCT) project COBOTIS (PTDC/EME-EME/32595/2017), and the Portugal 2020 project DM4Manufacturing POCI-01-0145-FEDER-016418 by UE/FEDER through the program COMPETE2020.

REFERENCES

[1] Grigory Antipov, Sid-Ahmed Berrani, Natacha Ruchaud, and Jean-Luc Dugelay. Learned vs. Hand-Crafted Features for Pedestrian Gender Recognition. In *ACM International Conference on Multimedia*, pages 1263–1266, 2015.

[2] M Blum, Jost Tobias Springenberg, J Wülfing, and M Riedmiller. A Learned Feature Descriptor for Object Recognition in RGB-D Data. In *IEEE International Conference on Robotics and Automation*, pages 1298–1303, 2012.

[3] Liefeng Bo, Xiaofeng Ren, and Dieter Fox. Unsupervised Feature Learning for RGB-D Based Object Recognition. In Jaydev P Desai, Gregory Dudek, Oussama Khatib, and Vijay Kumar, editors, *Experimental Robotics. Springer Tracts in Advanced Robotics*, pages 387–402. Springer, Heidelberg, 2013.

[4] Alexander Broad and Brenna Argall. Geometry-Based Region Proposals for Accelerated Image-Based Detection of 3D Objects. In *RSS Workshop on Deep Learning*, 2016.

[5] B Calli, A Walsman, A Singh, S Srinivasa, P Abbeel, and A M Dollar. Benchmarking in Manipulation Research: Using the Yale-CMU-Berkeley Object and Model Set. *IEEE Robotics & Automation Magazine*, 22(3):36–52, 2015.

[6] A Eitel, J T Springenberg, L Spinello, M Riedmiller, and W Burgard. Multimodal deep learning for robust RGB-D object recognition. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 681–687, 2015.

[7] A E Johnson and M Hebert. Using Spin Images for Efficient Object Recognition in Cluttered 3D Scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(5):433–449, 1999.

[8] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *International Conference on Neural Information Processing Systems*, pages 1097–1105, 2012.

[9] K Lai, L Bo, X Ren, and D Fox. A Large-Scale Hierarchical Multi-View RGB-D Object Dataset. In *IEEE International Conference on Robotics and Automation*, pages 1817–1824, 2011.

[10] K Lai, L Bo, X Ren, and D Fox. Sparse Distance Learning for Object Recognition Combining RGB and Depth Information. In *IEEE International Conference on Robotics and Automation*, pages 4007–4013, 2011.

[11] Y Lecun, L Bottou, Y Bengio, and P Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[12] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521:436–444, 2015.

[13] D G Lowe. Object Recognition from Local Scale-Invariant Features. In *IEEE International Conference on Computer Vision*, volume 2, pages 1150–1157, 1999.

[14] Nuno Mendes, Pedro Neto, J. Norberto Pires, and Altino Loureiro. Discretization and fitting of nominal data for autonomous robots. *Expert Systems with Applications*, 40(4):1143 – 1151, 2013.

[15] M M Rahman, Y Tan, J Xue, and K Lu. RGB-D object recognition with multimodal deep convolutional neural networks. In *IEEE International Conference on Multimedia and Expo*, pages 991–996, 2017.

[16] J Redmon and A Angelova. Real-Time Grasp Detection Using Convolutional Neural Networks. In *IEEE International Conference on Robotics and Automation*, pages 1316–1322, 2015.

[17] R B Rusu, G Bradski, R Thibaux, and J Hsu. Fast 3D Recognition and Pose Using the Viewpoint Feature Histogram. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2155–2162, 2010.

[18] M. A. Simao, P. Neto, and O. Gibaru. Unsupervised gesture segmentation by motion detection of a real-time data stream. *IEEE Transactions on Industrial Informatics*, 13(2):473–481, April 2017.

[19] A Singh, J Sha, K S Narayan, T Achim, and P Abbeel. BigBIRD: A Large-Scale 3D Database of Object Instances. In *IEEE International Conference on Robotics and Automation*, pages 509–516, 2014.

[20] Federico Tombari, Samuele Salti, and Luigi Di Stefano. Unique Signatures of Histograms for Local Surface Description. In *Computer Vision - ECCV 2010*, pages 356–369, 2010.