RGB-D Recognition and Localization of Cases for Robotic Depalletizing in Supermarkets

Pierluigi Arpenti[1], Riccardo Caccavale[1], Gianmarco Paduano[1], Andrea Fontanellli[1], Vincenzo Lippiello[1], Luigi Villani[1], and Bruno Siciliano[1]

***Abstract*—Integrating a robotic system into the depalletizing process of a supermarket demands a high level of autonomy, based on strong perceptive capabilities. This paper presents a system for detection, recognition, and localization of heterogeneous cases in a depalletizing robotic cell, using a single RGB-D camera. Such a system integrates apriori information on the content of the pallet with data from the RGB-D camera, exploiting a sequence of 2D and 3D model-based computer-vision algorithms. The effectiveness of the proposed methodology is assessed in an experiment where multiple cases and pallet configurations are considered. Finally, a complete depalletizing process is shown.**

***Index Terms*—Logistics, Object Detection, Segmentation and Categorization, Computer Vision for Automation.**

I. Introduction

Robotic logistics is the application of industrial robotics to optimize the flow of goods inside both manufacturing and logistics domains. Technological challenges are open for several scenarios, spreading from the unloading/loading to the palletizing/depalletizing of goods, as reported in [1]. In particular, depalletizing is the process of unloading an object, such as a corrugated carton on a pallet, in a defined pattern. This procedure is particularly common in logistics where the goods are typically delivered inside cases, which can be of different dimensions or weights. This task is a hard and tiresome activity for human workers since they have to manually remove a huge number of weighty cases.

Robotic depalletizing solutions (see Fig. 1) increase productivity and are often deployed inside factories. In many industrial contexts robotic depalletizing cells are typically tailored to specific products (having the same shape, same dimensions, and same appearance) and specific product lines. From the perception viewpoint, the depalletizing process results to be simplified, requiring a less amount of information. Examples of perceptual systems for structured depalletizing processes are proposed in [2], [3] where time of flight sensors [2] or combined RFID data and depth images [3] are deployed to recognize homogeneous cases. Analogously, 3D-vertices detection of cases is faced via edge detection and robust line fitting in [4]. On the other hand, in logistic scenarios like supermarkets, the products are heterogeneous (different shapes and dimensions) and are stored in *mixed pallets* (see Fig. 2). Therefore, depalletizing systems often require expensive or invasive sensors to recognize and localize products. For instance, in a complementary application, the palletization of cargo boxes, a strategy that uses a multi RGB-D system to extract the pose of boxes with sub-centimeter accuracy is proposed [5]. To avoid a slowdown in the depalletizing operations, the computer-vision algorithms often require strong modeling effort. For example, a system for real-time object detection and localization, based on a multi-resolution surfel model approach, is proposed in [6], whereas in [7] a low-cost photonic mixing device is combined with a predetermined model of loading configurations of the pallet. If it is not required to recognize the cases, the perception task reduces only to detection/localization. This situation is considered, for example, in [8], where the detection, the localization, and even the tracking of pallets transported by autonomous mobile robots are achieved through a Convolutional Neural Networks classifier. In applications like that of supermarkets, there are some products *(textured)* that exhibit faces covered by images and writings (illustrations regarding the goods contained by the case, recommendations about the storing and the opening modalities), while others *(untextured)* are completely blank.

This work was supported by EU H2020 REFILLS project under grant agreement 731590.

[1] Department of Electrical Engineering and Information Technologies, Universitá di Napoli Federico II, Italy name.surname@unina.it

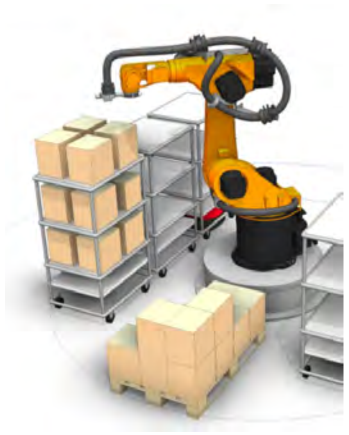


Fig. 1. Robotic depalletizing cell.

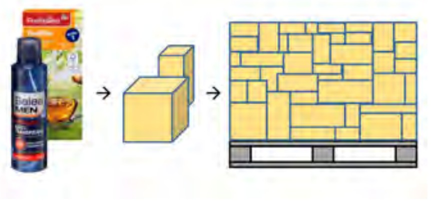


Fig. 2. Mixed pallet, made by products displaced in cases of different shapes.

For this latter class of packages, in [9] it is proposed the use of target planes extraction from depth images and packages border detection via brightness images to recognize various package stacked complicatedly. A similar perception system can be found also in [10], where a deep-learning approach that combines object detection and semantic segmentation has been applied to pick bins in cluttered warehouse scenarios. In this case, a specific data-reduction method is deployed to reduce the dimension of the dataset but several images of objects are still needed, impairing its usage by non-expert operators.

This work aims to design a simple framework for localization, detection, and recognition of cases from a mixed pallet, that can be easily integrated and deployed within robotic depalletizing systems. In particular, the proposed approach ensures: (1) high precision in the localization of cases, (2) non-invasive hardware requirements, and (3) a small input dataset (a single image for each face of the cases) that can be suitably collected by non-expert end-users. To this end, the designed system can detect, recognize, and localize cases of different types, both textured and untextured, from single-camera RGB-D images.

II. Detection, Recognition and Localization

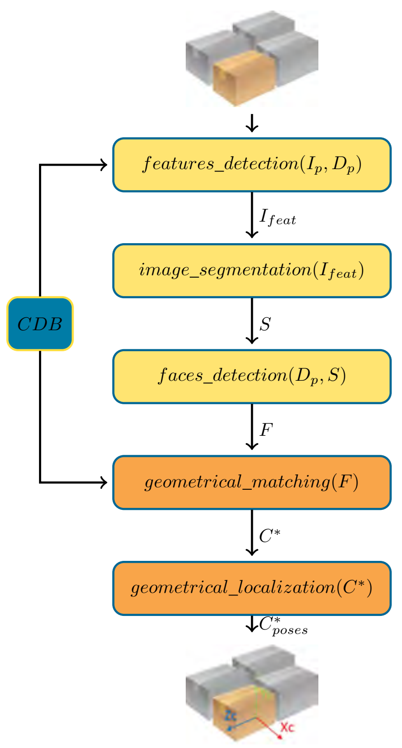
The presented system has been developed to allow a robotic cell to autonomously depalletize a mixed pallet, such as the ones that, nowadays, are manually depalletized by the human clerks in the backrooms of supermarkets. This task requires the capability to identify cases with different shapes and textures, as well as the capability to localize them in a fixed reference frame with a precision below a given tolerance, to use the estimated poses as references for the motion planner of the robot. The overall architecture (Fig. 3) is composed of three modules: *the cases database* (CDB) which contains the information about cases in the mixed pallet; *the detection module* (DM) where cases are detected from RGB-D data; *the geometrical module* (GM) which identifies such cases with those in the CDB and localizes them in the scene.

A. The Cases Database

The CDB contains information about the cases in the current mixed pallet. Each case c in the CDB is associated with a tuple $(b_c, n_c, x_c, y_c, z_c, \Gamma_c)$ where b_c is the product barcode, n_c is the number of instances of c in the pallet, x_c, y_c, and z_c are the dimensions of the case, and $\Gamma_c = (I_{c1}, I_{c2}, ..., I_{cn})$ is a set of images, one for each face of the case, if c is textured, otherwise it is empty. To standardize dimensions of each case, it is defined a convention where the z axis is always aligned to the depth of the case (see Fig. 4).

B. The Detection Module

The DM detects closed planar surfaces in the three-dimensional space, using both the RGB image I_p and the depth image D_p of the mixed pallet. It produces a preliminary association between regions in the space and the faces of the cases. The process can be divided into several distinct phases. As a preliminary step, since the position and the



$features\_detection(I_p, D_p)$
I_{feat}
$image\_segmentation(I_{feat})$
S
$faces\_detection(D_p, S)$
F
$geometrical\_matching(F)$
C^*
$geometrical\_localization(C^*)$
C^*_{poses}
CDB
Zc
Xc

Fig. 3. Overall architecture: CDB (blue), DM (yellow), and GM (orange).

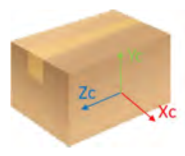


Fig. 4. Case reference frame: The axes are aligned to the orthogonal edges of the case.

dimensions of the pallet are known in advance, the background is removed from both D_p and I_p. Afterwards, a *feature detection* algorithm is firstly exploited to describe and detect local features, such as points, corners, edges, and so on, in I_p. Such features are then matched with those detected in I_c for each textured case c contained in CDB. Since multiple instances of the same case could be present in CDB ($n_c > 1$), two strategies to infer good matches are deployed. Firstly, given a set of matching features (keypoints) $m_1, m_2, \ldots, m_n$ between I_c and I_p, that are ordered by the Lowe's distance (refer to [11] for additional details), a subset of good matches $m_1, m_2, \ldots, m_k$ with $k < n$ is selected if and only if

$$\exists k < n, d_{lw}(m_k) < th\ d_{lw}(m_{k+1}) \qquad (1)$$

where d_{lw} is the distance of Lowe while $th \in [0, 1]$ is a suitable threshold, empirically set to 0.7, which is chosen so as to exclude matches whose Lowe's distance strongly differs from the distance of the previous ones. In this way, since a single feature can be assigned to multiple occurrences of the same case in I_p, the k-best features are collected instead of the single best one. These additional keypoints are virtually associated with the multiple instances of the cases. To distinguish such instances, a clustering process is used for both images (the case and the pallet one) and the comparison is made considering clusters of matching features belonging to the same region. In particular, multiple occurrences of the same case can lead to multiple clusters, then only the best cluster is selected, whose features likely belongs to a single instance of the case. If such features are in a sufficient number, then a homography transformation between the image I_c and the pallet image I_p is computed. In this case, the system creates a binary segment s_{feat} as a binary mask image corresponding to the region occupied by the image I_c of the matched face in the starting image I_p. The mask is then applied to D_p to isolate the region of the depth-map corresponding to s_{feat}. Moreover, since the segment s_{feat} is supposed to cover a single face, which is a region whose depth values are constant or change with continuity, an abrupt variation of the depth values indicate that the generated segment is not correctly fitting the face. For instance, if a textured case (in the background) is partially occluded by another case (in the foreground) and the number of matching keypoints is sufficient to cross the threshold, the occluded face can be detected by the algorithm. In this case, a new segment s_{feat} is generated, which does not correspond to a single face and must be discarded by the rest of the procedure. To do this, for each s_{feat}, the corresponding segment s_d is analyzed. If the discontinuity area is less than a given percentage threshold value of the total area of the depth segment s_d, than the segment s_{feat} is stored, otherwise, it is rejected. In the following experiments, such threshold has been experimentally set to 10 %, taking into account possible noisy measures of the RGB-D sensor. The segments which pass such depth test are stored in the set S_{feat} and finally used to create monochromatic patches which are added to the input image I_p to cover the textured cases detected, generating the image I_{feat}.

The whole process of feature detection is described in Algorithm 1. For each case c in CDB and for each image I_c in Γ_c, the matching features detected in I_c and I_p are collected into the set M (line 5). For each pair of features (m_i, m_{i+1}) in the set, the condition (1), necessary to collect the k-best features, is verified and the features are then stored in M^* (lines 6 to 8). Matches in M^* are associated to keypoints both in I_c and I_p. They are then clustered by Euclidean distance in order to find groups of features belonging to the same region. The sets K_p and K_c are created (line 12), where K_p contains all the clusters of keypoints found in I_p while K_c refers to I_c. In particular, $|K_c|$ is the number of clusters in which the keypoints belonging to a single face image I_c are grouped in. If the number of clusters in the input image, $|K_p|$, is greater than $|K_c|$, there are probably more instances of the same case inside the pallet. Anyway, the cluster with the highest number of keypoints in K_p and the corresponding cluster in K_c are collected in K_{match} (line 13). If the number of matches in K_{match} is sufficient (line 14), then the homography h between the two clusters of features is computed in the function $find\_homography()$ (line 15) and the binary segment s_{feat} is created from I_c through the function $binary\_segmentation()$ (line 16). If the corresponding depth region is flat, i.e., there are not huge

Algorithm 1 The feature detection procedure is invoked to take into account textured faces inside the pallet.

```
1:  procedure $features\_detection(I_p, D_p)$
2:      $I_{feat} = I_p$
3:      for each $c \in CDB$ do
4:          for each $I_c \in \Gamma_c$ do
5:              $M = feature\_matching(I_p, I_c)$
6:              for each $(m_i, m_{i+1}) \in M$ do
7:                  if $d_{lw}(m_i) < th\ d_{lw}(m_{i+1})$ then
8:                      $M^* = (m_1, \ldots, m_i)$
9:                      break
10:                 end if
11:             end for
12:             $(K_c, K_p) = clusterize(M^*, I_c, I_p)$
13:             $K_{match} = best\_match(K_c, K_p)$
14:             if $|K_{match}| \geq 4$ then
15:                 $h = find\_homography(I_c, I_p, K_{match})$
16:                 $s_{feat} = binary\_segmentation(h, I_c)$
17:                 if $is\_flat(s_{feat}, D_p)$ then
18:                     $S_{feat} \leftarrow S_{feat} \cup \{s_{feat}\}$
19:                     $B_{feat} \leftarrow B_{feat} \cup \{b_c\}$
20:                     $I_{feat} = add\_patch(I_c, I_{feat}, h)$
21:                 end if
22:             end if
23:         end for
24:     end for
25:     return $(I_{feat}, S_{feat}, B_{feat})$
26: end procedure
```

discontinuities of depth values in the region of the depth image D_p associated to s_{feat} (line 17), the segment s_{feat} is stored in the set S_{feat} (line 18) and a monochromatic patch is added to the region corresponding to s_{feat} (line 20). At the end of this process the procedure returns a patched image I_{feat} , the set of segments S_{feat} and the set of barcodes B_{feat} (line 25).

In particular, the image I_{feat}, which is the image of the pallet with the detected textured cases covered by patches, is then exploited by an *image segmentation* algorithm to segment the remaining untextured faces. The outcome of this process is twofold: the first output is the image I_{seg}, which is the completely segmented representation of the pallet; the second output is the set of segments S_{seg}, which is added to the set S_{feat} to get the set $S = S_{feat} \cup S_{seg}$ of all the segments s detected in the input image I_p.

In the *faces detection* phase, the binary segments s are exploited as masks for the depth image D_p, resulting in a sequence of depth segments s_d, as many as the segments stored in S, collected in the set S_d. Each of such depth segments is then exploited to build an associated point cloud which represents the depth segments in the three-dimensional space for the camera frame. For each segment s_{pc} obtained from the point cloud, the planar surface model is estimated through an *iterative method*. Once the parameters describing each planar surface have been correctly retrieved, all the points belonging to each point cloud are projected into the associated plane. Finally, assuming that all the faces of the cases are rectangular, each planar point cloud is bounded by the minimum-area enclosing rectangle. Such rectangles represent the candidate faces f which have to be recognized by the GM.

Algorithm 2 The face detection procedure detects candidate faces in the three-dimensional space.

```
1: procedure $faces\_detection(D_p, S)$
2:     for each $s \in S$ do
3:         $s_d = apply\_mask(D_p, s)$
4:         $s_{pc} = get\_pointcloud(s_d, D_p)$
5:         $p = find\_plan(s_{pc})$
6:         $s_{ppc} = project(s_{pc}, p)$
7:         $f = bounding\_rect(s_{ppc})$
8:         if $s \in S_{feat}$ then
9:             $F \leftarrow F \cup \{(f, b_c)\}$
10:        else
11:            $F \leftarrow F \cup \{(f,$“unknown”$)\}$
12:        end if
13:    end for
14:    return $F$
15: end procedure
```

The whole process concerning plane estimation and candidate face detection is described in Algorithm 2. For each binary segment s in S, the procedure begins with the creation of a depth segment s_d using s to mask D_p (line 3). A point cloud segment s_{pc} is obtained (line 4) which is used to estimate the underlying planar surface model p (line 5). Then s_{pc} is projected on the plane p obtaining a planar point cloud segment s_{ppc} (line 6), hence the bounding rectangle which is the candidate face f is built (line 7). Finally, if the segment s is in S_{feat}, the associated face f is stored in F along with the barcode b_c (line 9), otherwise, the segment s is in S_{seg} so no barcode is available. In this case, the face f is stored in F associated to the “unknown” identifier (line 11), indicating that the face is detected but not recognized. Notice that, if all the cases are untextured, then the feature detection procedure is never invoked and, as a consequence, the DM reduces to image segmentation and faces detection phases only.

C. The Geometrical Module

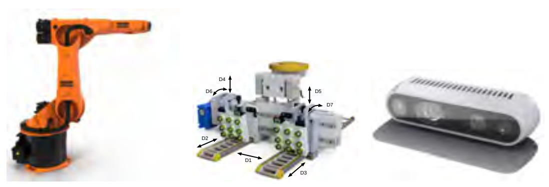
The role of GM is to associate each candidate faces $f \in F$ provided by the DM to one of the cases listed in the CDB and, successively, to estimate its pose. This task is addressed by two distinctive procedures: the *geometrical matching* and the *geometrical localization*. For each f stored in F with a barcode b_c, the geometrical matching checks if the width and the height characterizing f are compatible with two out of the three dimensions x_c, y_c, and z_c. The candidate face f is recognized, i.e. definitely associated with a case among those listed in the CDB, when the difference between its dimensions and those of the case is less than a given threshold value. Then, the matched case c^* can be stored in the set of the recognized cases C^*. Otherwise, if the dimensions are different, the geometrical matching function searches for the best matching case among the ones left in the CDB. If a good match is found, the matching case is stored in the set C^*. An analogous process is performed if f is associated to the “unknown” identifier, meaning that it has not been already associated with a barcode by the DM.

Algorithm 3 The geometrical matching procedure recognizes faces and associates them to the cases listed in the CDB.

```
1: procedure $geometrical\_matching(F)$
2:     for each $(f, b) \in F$ do
3:         if $b \neq$ “unknown” then
4:             $c^* = case\_from\_barcode(b)$
5:             if $match\_dimensions(f, x_{c^*}, y_{c^*}, z_{c^*})$ then
6:                 $c^* \leftarrow subs\_dimensions(f, x_{c^*}, y_{c^*}, z_{c^*})$
7:                 $C^* \leftarrow C^* \cup \{c^*\}$
8:             end if
9:         else
10:            $c^* \leftarrow find\_best\_match(f, CDB)$
11:            $c^* \leftarrow subs\_dimensions(f, x_{c^*}, y_{c^*}, z_{c^*})$
12:            $C^* \leftarrow C^* \cup \{c^*\}$
13:        end if
14:    end for
15:    return $C^*$
16: end procedure
```

The process of geometrical matching is described in Algorithm 3. For each candidate face (line 2), if f is associated with a known barcode b (line 3), the system selects from the database a case c^* having the same barcode (line 4). The dimensions of f and c^* are then compared (line 5) and, in case of matching, an updated description of c^* is stored in C^*. On the other hand, if the dimensions of c^* are too different from the detected one, then the procedure associate f with a new case c^* that better matches its dimensions (line 10). Also in this case, an updated version of c^* is stored in C^* (lines 11 to 13).

The cases recognized by the geometrical matching are subsequently localized by the geometrical localization, i.e. the pose of each matched case is estimated in a fixed reference frame (world frame). Such pose is described by the position of the centroid of the case and by the orientation of the reference frame relative to the case, which is the direction of the three coordinates axes aligned along with the cases orthogonal edges. Every matched case c^* is characterized by $\bar{c}_w = \begin{bmatrix} c_{w_x} & c_{w_y} & c_{w_z} \end{bmatrix}^T \in \mathbb{R}^3$, $\bar{c}_h = \begin{bmatrix} c_{h_x} & c_{h_y} & c_{h_z} \end{bmatrix}^T \in \mathbb{R}^3$, and $\bar{c}_d = \begin{bmatrix} c_{d_x} & c_{d_y} & c_{d_z} \end{bmatrix}^T \in \mathbb{R}^3$ which are the vectors (linked to the width, the height and the depth of the case, respectively) whose directions and norms describe the sought poses. The first two, at this stage of the process, are completely known, (they are the vectors associated to the matched face) whereas the direction of the third one has to be computed (the norm is equal to the dimension of the case which has not been matched). Such direction is given by the cross product $\hat{c}_d = \hat{c}_w \times \hat{c}_h$, where $\hat{c}_d$ is the unit vector of $\bar{c}_d$, $\hat{c}_w = \bar{c}_w / \|\bar{c}_w\|$ is the unit vector of $\bar{c}_w$, and $\hat{c}_h = \bar{c}_h / \|\bar{c}_h\|$ is the unit vector of $\bar{c}_h$. The orientation of the case is hence given by the matrix $\bar{\bar{c}}_o = \begin{bmatrix} \hat{c}_w & \hat{c}_h & \hat{c}_d \end{bmatrix} \in \mathbb{R}^{3 \times 3}$. On the other hand, the position of the centroid of the case, $\bar{C}_p \in \mathbb{R}^3$, is defined by projecting the center of the recognized face, $\bar{f}_p$, along $\hat{c}_d$, by a length



(a) KUKA KR60-3. (b) Gripper. (c) RealSense.

Fig. 5. Experimental Setup.

equal to the half the norm of $\bar{c}_d$, as:

$$\bar{c}_p = \bar{f}_p + \hat{c}_d \frac{\|\bar{c}_d\|}{2}. \tag{2}$$

Algorithm 4 The geometrical localization procedure localizes cases in the three-dimensional space.

```
1: procedure $geometrical\_localization(C^*)$
2:     for each $c^* \in C^*$ do
3:         $\bar{\bar{c}}_o = get\_orientation(\hat{c}_w, \hat{c}_h)$
4:         $\bar{c}_p = get\_position(\bar{f}_p, \bar{c}_d)$
5:         $c^*_{pose} = (\bar{c}_p, \bar{\bar{c}}_o)$
6:         $C^*_{poses} \leftarrow C^*_{poses} \cup \{c^*_{pose}\}$
7:     end for
8:     return $C^*_{poses}$
9: end procedure
```

The localization process is described in Algorithm 4. For each recognized case c^*, the orientation is given by the unit vectors $\hat{c}_w$, $\hat{c}_h$, and $\hat{c}_d$ where the third one is computed through the cross product (line 3). Known the three unit vectors, as well as the three dimensions of the case, it is possible to estimate the position of the centroid as in (2) (line 4). Known c^*_p and c^*_o, the pose c^*_{pose} of each recognized case is hence available (line 5). Finally, the outcome of the whole procedure is the set C^*_{pose} of all the poses of the recognized cases.

III. Case Studies

This section presents two case studies on a robotic cell developed in a real supermarket's backroom environment which includes (see Fig. 5): a 6-DoF KUKA KR60-3 manipulator with 2-meter radius workspace; a reconfigurable gripper adaptable to cases of different dimensions; an RGB-D camera placed between the robot and the pallet, front-facing the latter. The cell is also endowed with an executive system [12] in charge of scheduling and regulating the execution of the robotic depalletizing tasks. The RGB-D camera is the RealSense D435, a USB-powered depth camera that consists of an RGB sensor, a pair of depth sensors, and an infrared projector. It was selected due to the high output resolution guaranteed both for RGB and depth data, up to 1280 x 720 p. The camera has been fixed to the base of the KUKA KR60-3, in front of the mixed pallet, at a distance of $\simeq$ 1.5 m. The camera pose is assumed to be known respect to the fixed reference frame, through a calibration procedure. The mixed pallet is made of nine cases, seven textured, and the remaining two untextured. Next to the robot, outside the camera field of view, a trolley with three shelves hosts the depalletized cases.

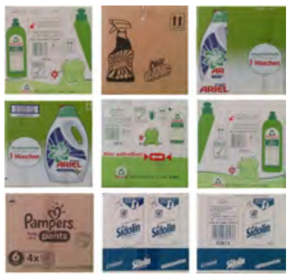


Fig. 6. Some pictures of the textured faces stored in the CDB.

A. Case Study I: Detection, Recognition and Localization

The first case study aims at evaluating the recognition and localization capabilities of the system. 9 cases (see Fig. 6 for an example of textured cases) organized in 10 different configurations of the mixed pallet, obtained by changing the positions of the cases, have been tested. For each configuration, the system was executed 3 times, for a total of 180 recognitions and localizations to be performed. The actual pose of each box is known because it was computed employing markers that have been displayed on every single case in a preliminary step. The experiments carried out show that 177 faces out of 180 were correctly identified, with an accuracy of $\simeq$ 98 %. The system failed 3 times: 2 times visible cases were not recognized correctly, while 1 time an occluded face was recognized.

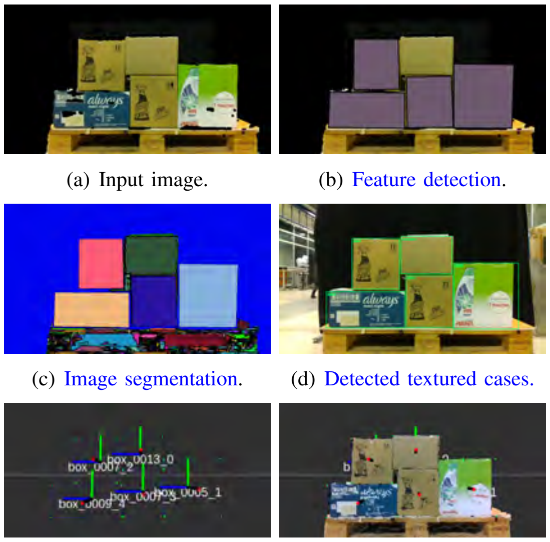
Table III-A, which refers only to the recognized cases, shows that the proposed system guarantees a good localization performance, with precision under the tolerance imposed by the gripper which requires a maximum estimation error of 0.020 m on the axes y_c.

	x (m)	y (m)	z (m)
Avg. Err.	-0.00179	-0.002	-0.012
Norm. Avg. Err.	0.010	0.008	0.012
Std. Dev.	0.013	0.010	0.007

TABLE I
Statistics of the centroids position estimation error

B. Case Study II: the Autonomous Depalletization

This case study shows the whole depalletizing process (see Fig. 7 and Fig. 8). For the test, a pallet configuration that contains two instances of the same case was selected. The experiments were carried out in a cluttered environment without structured light. To avoid misreadings of the RGB-D sensor due to reflections, a black tendon was positioned behind the pallet. As explained in Section II, the input image is firstly segmented by a feature detection algorithm. In this experiment, *Scale Invariant Feature Transform (SIFT)* [11] was chosen, due to its robustness. The resulting patched image is then segmented through the *Watershed Transform* [13] which is an image segmentation method, based on intensity images, well suited for monochromatic faces. The segments which result from these two processes are then converted into binary images which in turn lead to planar surfaces estimated with *Random Sampling and Consensus (RANSAC)*



(a) Input image. (b) Feature detection.

(c) Image segmentation. (d) Detected textured cases.

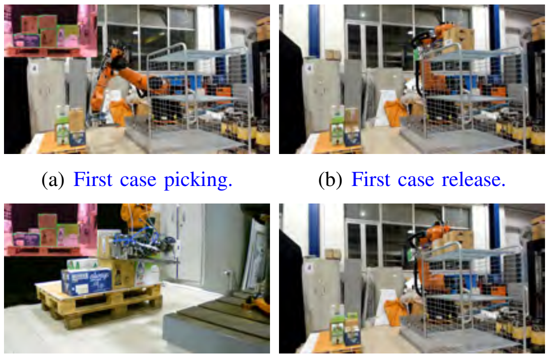
(e) Estimated output poses. (f) Cases represented as point cloud match with the poses.

Fig. 7. System workflow. At each iteration, the faces in the input image (a) are detected via SIFT (b) and Watershed (c), and the textured ones are preliminary associated to barcodes (d). The geometrical matcher associates the cases in the CDB to each detected face and then the geometrical localizer estimates their poses (e, f).

[14] and hence to candidate faces in the three-dimensional space. The information contained in the CDB is then compared to those associated with each candidate face and finally, the cases are definitively recognized and localized in a fixed reference frame. The estimated poses are sent to the high-level executive system that, according to a given policy, regulates the task execution. The system decomposes high-level tasks into low-level motion primitives while robot motion planning is performed via ROS MoveIt. Each depalletizing task is composed of a recognition phase where the designed system is deployed to detect/recognize the cases, providing also their pose, a picking phase where one of the perceived case is taken from the mixed pallet (following the policy provided by the executive system), and a release phase where the case is stored in a specific location on a trolley. When a depalletizing task is accomplished, the stored case, with all the related information, is removed from the CDB and the whole process starts again from the beginning. The whole test has been completed, with all 9 cases of the mixed pallet correctly depalletized.

IV. Conclusions

In this paper, a novel RGB-D recognition and localization methodology for the depalletizing of cases in supermarkets has been proposed. The approach has two main advantages: it relies on a short dataset where only one image for each face of the cases is needed, and it is based on a single image from a single RGB-D sensor, hence supporting its integration into preexisting logistics scenarios. As demonstrated via real experiments, such methodology effectively tackles the mixed pallet depalletization, i.e. the task of unloading cardboard cases of different types. As future work, an exhaustive comparison in terms of segmentation accuracy and depalletization speed with other methods (convolutional neural networks and support vector machines, for example) will be performed. Moreover, a multi-camera approach will be explored to understand if multiple images from different perspectives can lead to a significant improvement in estimating the poses of the cases and if it is worth the extra complexity and cost.



(a) First case picking. (b) First case release.

(c) Second case picking. (d) Second case release.

Fig. 8. Picking and release phases for two consecutive depalletized cases.

References

[1] W. Echelmeyer, A. Kirchheim, and E. Wellbrock, "Robotics-logistics: Challenges for automation of logistic processes," in *IEEE Int. Conf. on Automation and Logistics*, 2008, pp. 2099–2103.

[2] D. Katsoulas and D. I. Kosmopoulos, "An efficient depalletizing system based on 2d range imagery," in *IEEE Int. Conf. on Robotics and Automation*, 2001, pp. 305–312.

[3] C. Prasse and S. Skibinski and F. Weichert and J. Stenzel and H. Müller and M. ten Hompel, "Concept of automated load detection for de-palletizing using depth images and RFID data," *IEEE Int. Conf. on Control System, Computing and Engineering*, 2011, pp. 249–254.

[4] D. Katsoulas and B. Lothar, "Efficient 3D Vertex Detection in Range Images Acquired with a Laser Sensor," in *23rd DAGM-Symposium on Pattern Recognition*, 2001, pp. 116–123.

[5] Y. Raaj, S. Nair, and A. Knoll, "Precise Measurement of Cargo Boxes for Gantry Robot Palletization in Large Scale Workspaces Using Low-Cost RGB-D Sensors," in *Asian Conf. on Computer Vision*, 2016, pp. 472–486.

[6] D. Holz, A. Topalidou-Kyniazopoulou, J. Stückler, and S. Behnke, "Real-time object detection, localization and verification for fast robotic depalletizing," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2015, pp. 1459–1466.

[7] F. Weichert, S. Skibinski, and J. e. a. Stenzel, "Automated detection of euro pallet loads by interpreting pmd camera depth images," *Logistics Research*, vol. 6, pp. 99–118, 2013.

[8] I. Mohamed, A. Capitanelli, and F. e. a. Mastrogiovanni, "Detection, localisation and tracking of pallets using machine learning techniques and 2D range data," *Neural Computing and Applications*, vol. 32, pp. 8811–8828, 2019.

[9] H. Nakamoto, H. Eto, T. Sonoura, J. Tanaka, and A. Ogawa, "High-speed and compact depalletizing robot capable of handling packages stacked complicatedly," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2016, pp. 344–349.

[10] M. Schwarz, A. Milan, A. S. Periyasamy, and S. Behnke, "RGB-D object detection and semantic segmentation for autonomous manipulation in clutter," *Int. J. of Robotics Research*, vol. 37, pp. 437–451, 2018.

[11] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int, J. of Computer Vision*, vol. 60, pp. 91–110, 2004.

[12] R. Caccavale and A. Finzi, "Flexible task execution and attentional regulations in human-robot interaction," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 9, pp. 68–79, 2016.

[13] F. Meyer, "Color image segmentation," in *Int. Conf. on Image Processing and its Applications*, vol. 2, 1992, pp. 303–306.

[14] M. A. Fischler and R. C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," *Communication of the ACM*, vol. 24, no. 6, 1981.