REGULAR PAPER

Vision and Tactile Robotic System to Grasp Litter in Outdoor Environments

Ignacio de Loyola Páez-Ubieta[1,2] · Julio Castaño-Amorós[1] · Santiago T. Puente[1] · Pablo Gil[1]

Abstract

The accumulation of litter is increasing in many places and is consequently becoming a problem that must be dealt with. In this paper, we present a manipulator robotic system to collect litter in outdoor environments. This system has three functionalities. Firstly, it uses colour images to detect and recognise litter comprising different materials. Secondly, depth data are combined with pixels of waste objects to compute a 3D location and segment three-dimensional point clouds of the litter items in the scene. The grasp in 3 Degrees of Freedom (DoFs) is then estimated for a robot arm with a gripper for the segmented cloud of each instance of waste. Finally, two tactile-based algorithms are implemented and then employed in order to provide the gripper with a sense of touch. This work uses two low-cost visual-based tactile sensors at the fingertips. One of them addresses the detection of contact (which is obtained from tactile images) between the gripper and solid waste, while another has been designed to detect slippage in order to prevent the objects grasped from falling. Our proposal was successfully tested by carrying out extensive experimentation with different objects varying in size, texture, geometry and materials in different outdoor environments (a tiled pavement, a surface of stone/soil, and grass). Our system achieved an average score of 94% for the detection and Collection Success Rate (CSR) as regards its overall performance, and of 80% for the collection of items of litter at the first attempt.

Keywords Litter detection · Object recognition · Tactile sensing · Tactile learning · Grasping

1 Introduction

Several environmental problems currently harm our planet, one of which is the accumulation of waste such as plastic bottles, metal cans, drink cardboard or glass that is clearly visible in the streets and parks of cities. The mean estimated degradation time for cardboard and glass ranges from 5 to 4,000 years, respectively. In order to help avoid the contamination of soil and the environment, this waste should be selectively picked up in an automated manner for its subsequent recycling. We propose to solve the problem of collecting this kind of waste outdoors by providing a robotic system that incorporates several visual-tactile perception systems.

Several solutions that use robots for cleaning purposes already exist in literature. For instance, [1] solve the aforementioned problem in a simulated indoor environment, and [2] presents a re-configurable cleaning robot that works in a real environment, but these solutions do not provide object recognition or manipulation skills that enable refuse to be picked. In this line, [3] shows an outdoor solution in which a robot detects and picks up refuse bags, but without tactile perception or variability in the scenarios. At present, robot learning techniques make it possible to address interactions with the environment during navigation and manipulation tasks, as shown in [4]. Although its results are promising, this kind of approach requires a lot of training data and is usually limited to controlled settings, which are usually indoors.

The technique presented herein has been tested in real-world scenarios by performing not only detection, as occurs

✉ Ignacio de Loyola Páez-Ubieta
ignacio.paez@ua.es

✉ Julio Castaño-Amorós
julio.ca@ua.es

Santiago T. Puente
santiago.puente@ua.es

Pablo Gil
pablo.gil@ua.es

1 AUtomatics, RObotics and Artificial Vision Lab, University of Alicante, Alicante, Spain

2 University Institute for Engineering Research, Miguel Hernandez University, Elche, Spain

in [5], but also the recognition of a wider number of waste elements than occurs in [6]. This is done by extracting the best grasping points of the segmented point cloud, signifying that there are fewer points to process, thus lightening the computation time when compared to that of [7]. Unlike the aforementioned works, our approach also includes object handling skills thanks to new tactile perception algorithms, which make it possible to accomplish stable grasping when picking up litter.

Our main contributions are:

- We propose a tactile-based grasping estimation method for the picking of waste objects. We specifically use low-cost visual-based sensors for tactile manipulation in order to carry out the task of litter collection. As these tactile sensors do not have a mathematical equation with which to map tactile images (intensity) onto force in N. and they also do not contain visual markers to estimate the movement variation, then our data-based methods are, therefore, crucial as regards performing this task correctly.
- We generate two datasets, one containing tactile images for the contact and slip detection tasks, and another containing color images of household waste in a wide variety of outdoor environments, for the object detection and localisation tasks.
- We present comparative studies of object recognition and contact-slip detection during the grasping task, first using our datasets and later grasping litter in outdoor environments such as in our university campus. Therefore, another contribution is based on the design, implementation and communication of the different perception modules on a real robot system applied to the task of litter collection in outdoor environments.

Our robotic system is divided into two main modules: the vision module and the tactile module. The vision module is able to detect and recognise litter from images and to calculate grasping points from the point cloud of the object. The tactile module performs grasping detection and control on the basis of tactile feedback. Our solution has been integrated and tested using an UR5e commercial robotic arm with a 2F-140 ROBOTIQ gripper. The arm was installed on our mobile robotic platform with autonomous navigation, called BLUE [8], and can be seen in Fig. 1. BLUE has several sensors to which we have added IMU and RGBD cameras (Intel® RealSense™ D435i) for detection and recognition, along with two DIGIT sensors [9] attached to the fingertips of the gripper for the tactile operations.

This paper is organised as follows: Sect. 2 provides an explanation of related works regarding each part of the pipeline., while Sect. 3 shows a description of the proposed methods of which our visual-tactile perception system is composed. A description of how each of the methods is trained and validated separately is provided in Sect. 4, along with the preliminary results obtained after carrying out tests with previously unseen items of waste. The results obtained by the whole visual-tactile perception system in real environments are then presented, and finally, the paper concludes with a discussion of the results obtained and the performance of the proposal.

2 Related works

2.1 Visual perception

The rapid development of deep learning has led to an avalanche of object detection methods [10]. For example,
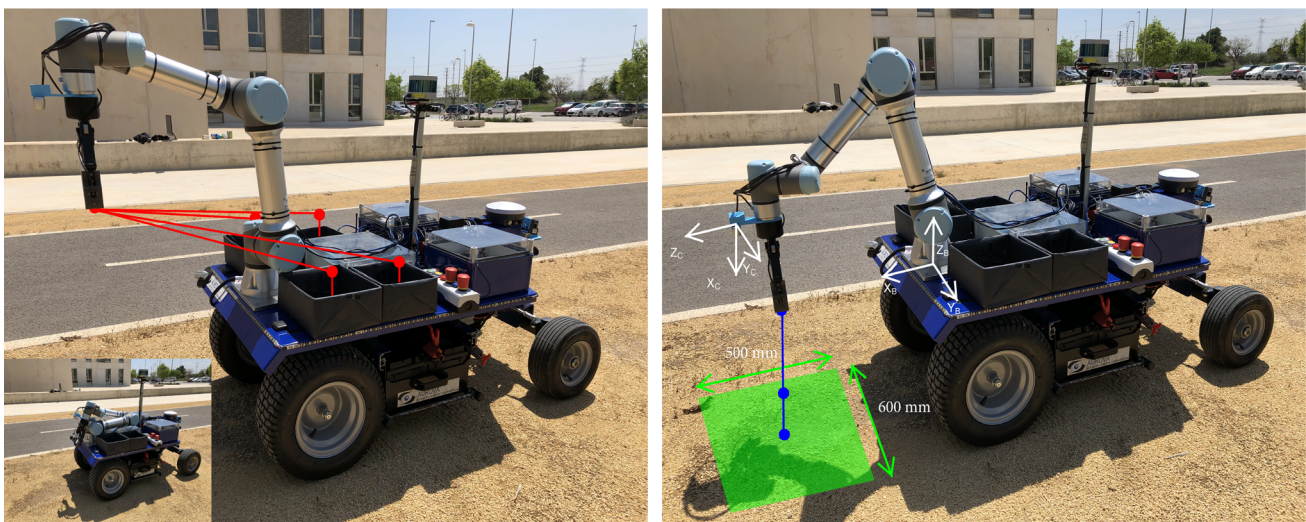

Fig. 1 Home, navigation and detection poses of BLUE robot. (Left) UR5e pose when BLUE is in home pose. (Bottom-left) Navigation pose when BLUE is moving around. (Right) Detection pose when BLUE is near an item of litter

[5] used an AlexNet Convolutional Neural Network (CNN) to perform image classification tasks in both indoor and outdoor environments. However, this work presents solely a detection system for household waste rather than a system with which to pick waste up.

Other works attempt to solve more complex problems, such as object detection and segmentation. In [11], the author uses two Neural Networks (NNs) to eliminate the ground and classify six kinds of waste items. We, however, use a single NN to locate and classify the objects, thus speeding up the process. Going one step further, [12] trained a YOLACT NN to classify items of domestic waste in indoor environments. They tested the model with only three categories ("plastic", "metal" and "cardboard"). We, in contrast, add a new class of glass objects, thus making the detection more difficult owing to their transparency. Moreover, our system works in outdoor environments in which lighting conditions cannot be controlled.

The waste collection task makes it necessary to address not only its detection and location but also the grasping of objects [13], as already implemented in the robotic field. For instance, [7] proposed a mathematical method with which to calculate the best pair of grasping points from a 3D-scene point cloud. This method extracts the object and obtains the grasping pose using the curvature of the object. In our work, we use a segmented point cloud of the 3D-object for the grasping task. Reducing the number of candidate points makes the process faster. Another NN approach, which is shown in [14], generates grasping proposals from voxelised 3D-object point clouds. But it has some limitations as regards complete outdoor point clouds rather than segmented object point clouds and working in real time. Our proposal is an improvement in this respect, since it is faster in outdoor scenarios.

2.2 Tactile perception

Several types of tactile sensors (capacitive, resistive, barometric, optical-based, etc.) have been developed in recent years in order to assist in robotic manipulation tasks such as contact or slip detection, which are essential for a safe grasp when picking up objects. In previous works, [15] calculated contact by comparing the intensity of the colour of two tactile images using an optical tactile sensor called Gelsight [16]. However, this method requires the readjustment of its parameters and is less robust to uncertainty owing to the use of traditional computer vision techniques. In [17], the authors trained a Recurrent Neural Network (RNN) to detect contact events in image sequences from another optical tactile sensor called FingerVision [18]. Nonetheless, this sensor contains markers that are complex to generate and require high-cost machinery. The DIGIT sensors do not contain markers, while

our contact detection CNN-based method is more robust than traditional computer vision techniques.

When employed in the literature related to this field, the term slip usually refers to the normal component of the grasping force going outside of the grasping cone. Detecting and reacting to this phenomenon is, therefore, fundamental if an object is to be grasped correctly. In [19] a Support Vector Machine (SVM) algorithm is trained in order to classify whether images obtained from a TacTip sensor are stable or slipping events. However, they do not grasp the object with a gripper or hand, but rather push the object against the wall in order to stop the slip movement. This application is, therefore, limited. The slip or stable classification task has also been studied in [20], in which visual (eye-to-hand camera) and tactile (Gelsight sensor) information is combined. However, it is necessary to process a lot of data, and real-time execution is not guaranteed. Conversely, our slip detector works extremely fast and allows the implementation of our controller. In a major advance, [21] researched the slip detection problem using a multi-fingered hand. Nonetheless, detecting slip is more complex as the number of sensors increase. Our approach of using a two-fingered robotic gripper is, therefore, more optimal for the litter collection task.

# 3 Our approach: Visual-tactile perception for robotic manipulation

In this work, we propose a visual-tactile perception system for robotic manipulation, whose architecture is shown in Fig. 2. Only the six DoFs manipulation arm of the mobile manipulator robotic system described in Sect. 1 is employed for the task of litter collection, assuming that the navigation task towards the litter objects is already carried out, thus obtaining an optimal localization of the mobile platform. Detailed descriptions of the three main modules of our system are provided in the following sections.

3.1 Object detection and localisation

In this section, the first module is explained, which consists of detecting and locating the item of litter using a CNN.

The first task carried out by our system is detecting and classifying litter in outdoor environments. In order to accomplish it, our BLUE robot performs the exploration task using several navigation algorithms. While the exploration is taking place, BLUE is able to annotate objects as possible instances of refuse and compute their spatial location in the world [22], thus making it possible to plan trajectories towards them. The visual sub-system, which is shown in the green scheme in Fig. 2, captures images while the robot is navigating and then processes them using a CNN to obtain the object position and category of the objects. After analysing several CNNs

Fig. 2 Scheme of our tactile-visual system for robotic grasping. It is made up of two main parts: waste detection and recognition (green part) and tactile perception in order to manipulate the item of waste (blue part)
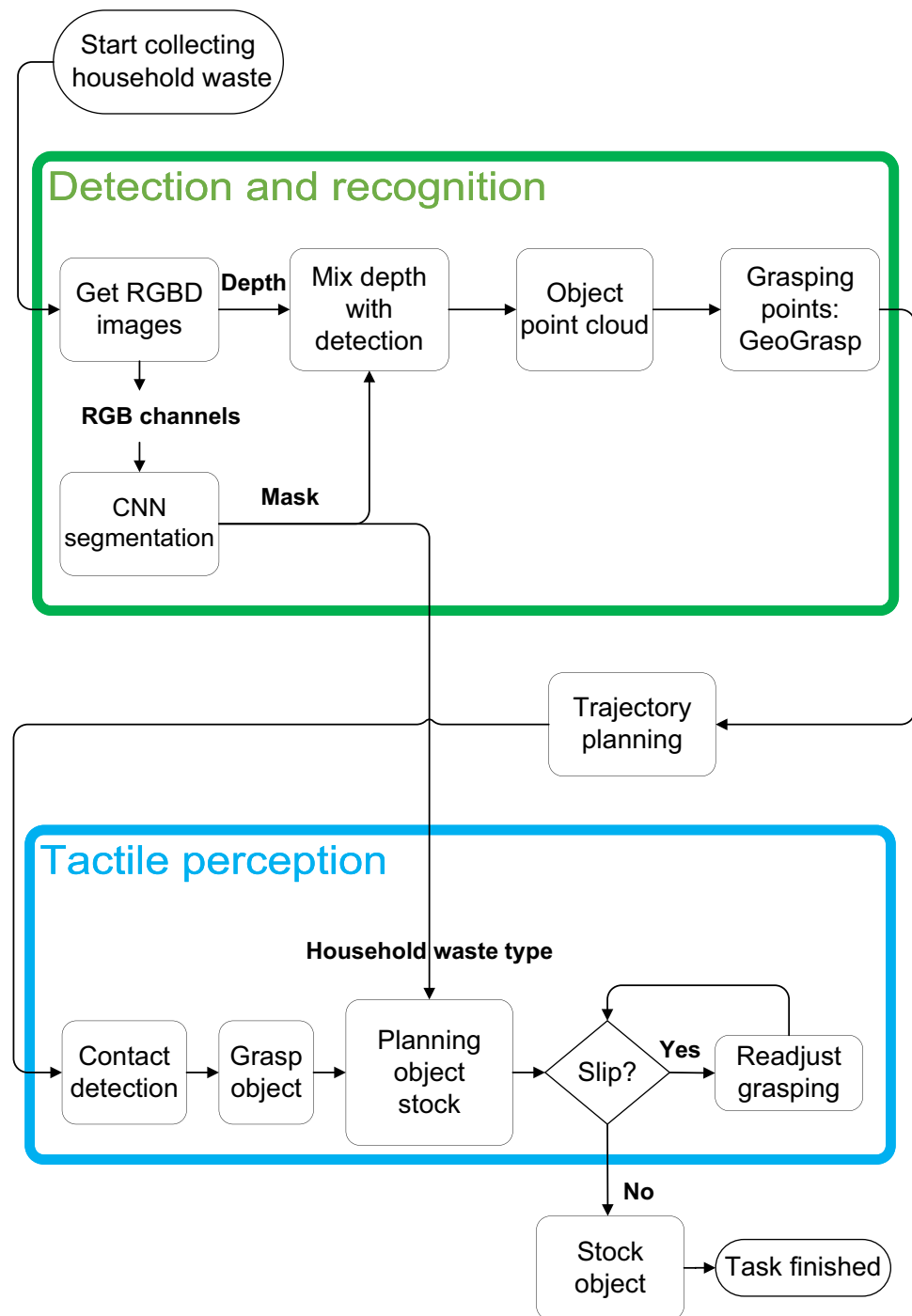
available in the state of the art [23], [24], we chose Mask R-CNN [25], YOLACT [26] and YOLACT++ [27]. These CNNs appear to work well in a wide variety of fields that require segmentation tasks, such as understanding natural scenes or intelligent driving. In the former, it improves object detection since it avoids some cases of occlusion by providing a more detailed analysis of the image, while in the latter it is used to determine the localisation of major categories of objects such as street lights or people running.

On the one hand, Mask R-CNN is considered to be a two-stage detector, since it has two parts. The first generates Regions of Interest (RoIs), and the second classifies and segments those RoIs. These detectors have drawbacks, such as low performance and a dependency on feature localisation. Its architecture is similar to that of the Faster-RCNN [28], since it adds a new layer to the Faster-RCNN in order to predict a segmented mask. In this case, there are three output layers: the class label, the Bounding Box (BB), and the
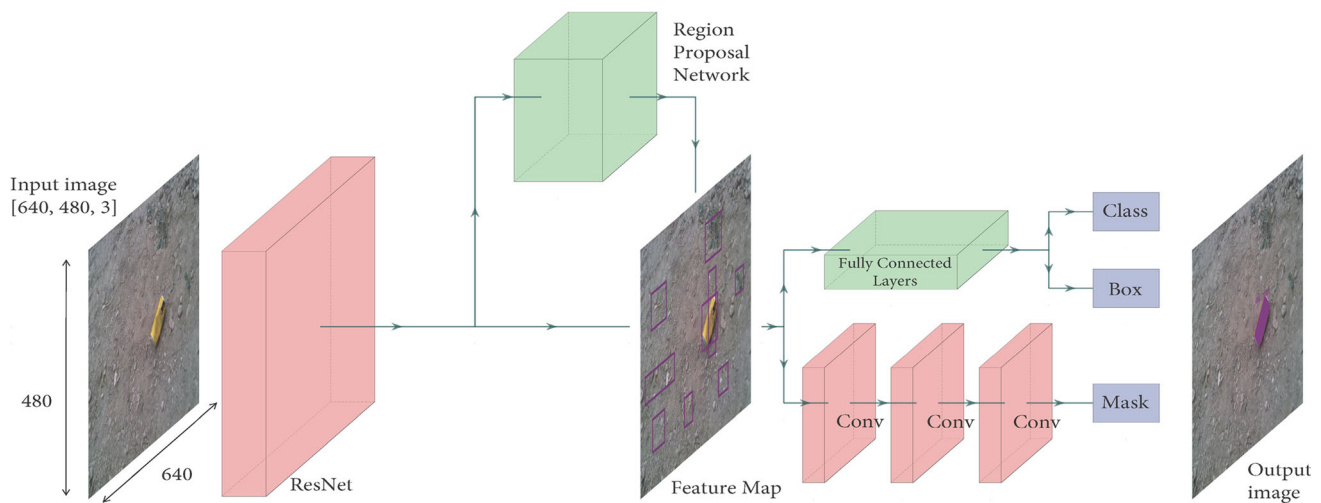
Fig. 3 Mask R-CNN architecture with an example of the input and output from the litter detection task

aforementioned segmented mask. These changes comprise an improved RoI Polling layer (RoI Align) and the addition of a segmented mask output layer in Mask R-CNN (see Fig. 3).

On the other hand, YOLACT is considered to be a single-stage detector that performs the instance segmentation in one step. This makes it really fast and allows it to achieve real-time inference. The detection task is divided into two simpler parallel sub-tasks. The first consists of the prototype generation branch, which predicts a k-set of prototype masks without loss (this loss exists in the traditional methods). The second involves the mask coefficient branch, which is a vector of mask coefficients for each prototype. These result in k-mask coefficients (one per prototype), c class confidences, and 4 BB regressors, producing a total of $4 + c + k$ coefficients per anchor. Finally, the two sub-tasks come together in the mask assembly process. This is done by applying a linear combination to both sub-tasks and a sigmoid non-linearity process (see Fig. 4). YOLACT++ was created later by making some minor changes to YOLACT. These changes range from a fast mask re-scoring network to deformable convolution in layers, including an optimised prediction head.

One of the configurable parameters in both CNNs is the backbone. Our visual sub-system is implemented in order to choose between the ResNet50-FPN and the ResNet101-FPN. Mask R-CNN includes the DarkNet53-FPN as an extra backbone. The number that accompanies each backbone determines the number of layers. In fact, all of these CNNs include a Feature Pyramid Network (FPN) as part of their architecture.

ResNet [29] stands for Residual Neural Network. In 2015, it was still believed that adding additional layers to a NN would make it work better. This worked in theory but not in practice since there was the problem of the vanishing gradient. This new architecture solved that problem by incorporating residual blocks with skip connections.

The FPN consists of reducing the size of an image step by step. Image features are then extracted from both the original and the scaled images in each step. FPN later combines all the features that have been extracted, mixing both low-resolution semantically strong and high-resolution semantically weak features. This combination can be achieved by using top-down and lateral connections, thus leading to better results in outdoor image analysis.

Another backbone, which is written in C and CUDA, is DarkNet [30]. This rose to fame because it improved the performance of ResNet101-FPN and carried out the detection process 1.5x faster.

In this work, we shall analyse the behaviour and performance of these architectures as backbones in detection tasks for robotic manipulation in real outdoor environments. We first present the validation results, which were obtained after running our system offline. That is, we used pre-recorded videos of previous navigation missions that had already been carried out (see Sect. 4.2). Finally, we show additional results in new scenarios. These have not been seen by our NN before and were captured in real-time navigation mode (see Sect. 4.5).

3.2 Grasping computing and trajectory planning

This section describes how our system estimates the grasping points from the 3D point cloud to collect the item of litter.

Once the object has been recognised and is in the robot's reachable workspace, our manipulator robot has to pick it up. It is, therefore, necessary to estimate the grasp. In our case, we obtain grasping points by using a method called GeoGrasp. It is based on geometry and needs a raw scene
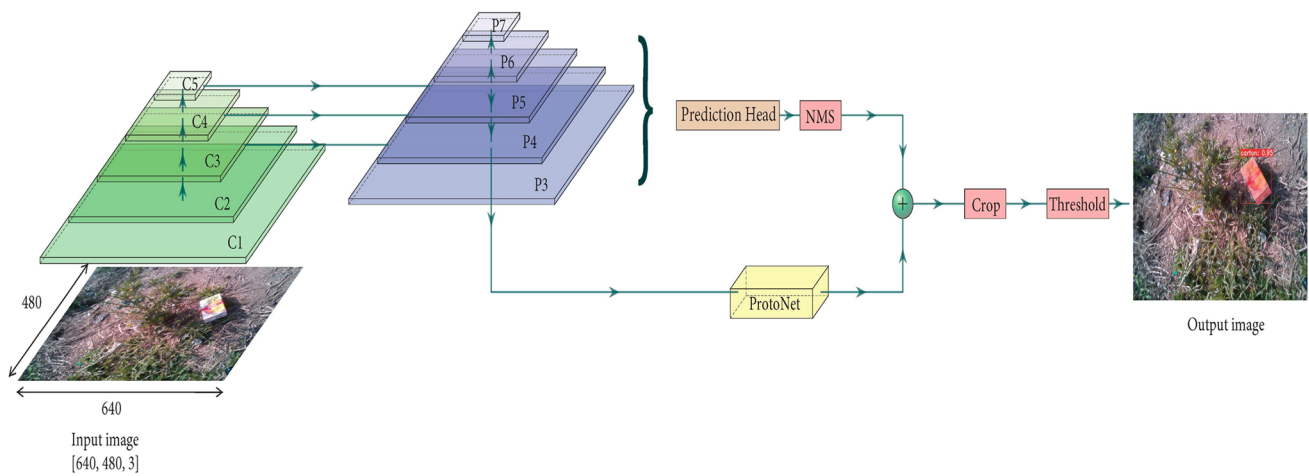
Fig. 4 YOLACT architecture with an example of the input and output from the litter detection task

point cloud as input. But in this work, the input has been changed, as inspired by [31]. A filtered point cloud is, therefore, employed.

An image is initially captured using a RealSense™ D435i depth camera. The depth image has a resolution of 640x480 pixels, which matches that of the RGB image. The RGB image is then processed by Mask R-CNN, YOLACT or YOLACT++. The detection task results in a cluster of pixels containing the object, a BB, and a category. This result allows the creation of a new point cloud that includes only the RGBD points considered by the NN as object points. Then, GeoGrasp calculates the grasping points in the new point cloud and locates these points in the 3D space referenced at the base of the robot. Two values are required for this task: the transformation from the proposed grasping points to the camera and the transformation from the camera to the robot base.

The first transformation is calculated using the proposed grasping points and the camera intrinsic parameters as in Eq. (1).

$$
\begin{bmatrix} x_C \\ y_C \\ z_C \end{bmatrix} = \begin{bmatrix} R^t \end{bmatrix} \begin{bmatrix} M_c \end{bmatrix}^{-1} \begin{bmatrix} x_G \\ y_G \\ 1 \end{bmatrix}
$$

$$
d = \underbrace{\begin{bmatrix} 0\ 0\ 1 \\ 1\ 0\ 0 \\ 0\ 1\ 0 \end{bmatrix}}_{\mathbf{R^t}} \underbrace{\begin{bmatrix} 1/\mathbf{f_x} & 0 & -\mathbf{c_x}/\mathbf{f_x} \\ 0 & 1/\mathbf{f_y} & -\mathbf{c_y}/\mathbf{f_y} \\ 0 & 0 & 1 \end{bmatrix}}_{\left[M_c \right]^{-1}} \begin{bmatrix} x_G \\ y_G \\ 1 \end{bmatrix} d \in \mathbb{N}_{\geq 0} \quad (1)
$$

where $[x_G, y_G]$ denote the coordinates of the proposed grasping point in a 2D image for the x and y axes respectively, (f_x, f_y) and (c_x, c_y) respectively denote the focal length and the camera center (obtained from the intrinsic parameters of the camera) in pixels, R^t denotes a consecutive rotation of -90 degrees in the x and y axes respectively, while M_c denotes the

calibration matrix (obtained by using ArUco markers [32]) and d denotes the depth of the point in mm (obtained from the D channel of the RGBD image captured).

The second transformation consists of transforming $[x_C, y_C, z_C]$ into the coordinates of the base of the robot by following Eq. (2).

$$
\begin{bmatrix} x_B \\ y_B \\ z_B \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R(q)} & \mathbf{p(q)} \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} ^E T_C \end{bmatrix} \begin{bmatrix} x_C \\ y_C \\ z_C \\ 1 \end{bmatrix} \quad (2)
$$

vskip-5ptwhere $[x_C, y_C, z_C]$ denote the 3D coordinates relative to the camera, $[x_B, y_B, z_B]$ denote the new coordinates relative to the base in meters, $\mathbf{q} \in \mathfrak{R}^6$ denotes a vector of joint coordinates, $\mathbf{p(q)} \in \mathfrak{R}^3$ denotes the position vector from the robot´s base to the coordinate frame (forward kinematics), $\mathbf{R(q)} \in SO(3)$ is the end-effector orientation and $^E T_C$ is the fixed transformation between the camera and the robot end-effector. These coordinate systems can be seen on the right-hand side of Fig. 1.

One drawback of using only the segmented object points is that we have to trust the NN detection since our calculation is based on its result. The worse the NN detection is, the worse the grasping points will be positioned. The full process is shown in Fig. 5.

Once the objects have been detected and localised, and their grasping estimated, the robotic arm mounted on the BLUE robot has to reach them. This is done using ROS Moveit! [33]. Moveit! provides the UR5e arm with a motion planning framework in order to compute and test the trajectories before operating the real physical robot. A wide variety of motion planning controllers are provided, but we use RRT* [34], [35] (an asymptotically optimal version of Rapidly-exploring Random Trees).

It is worth mentioning that some of the trajectories computed will always be the same. This has been borne in mind,
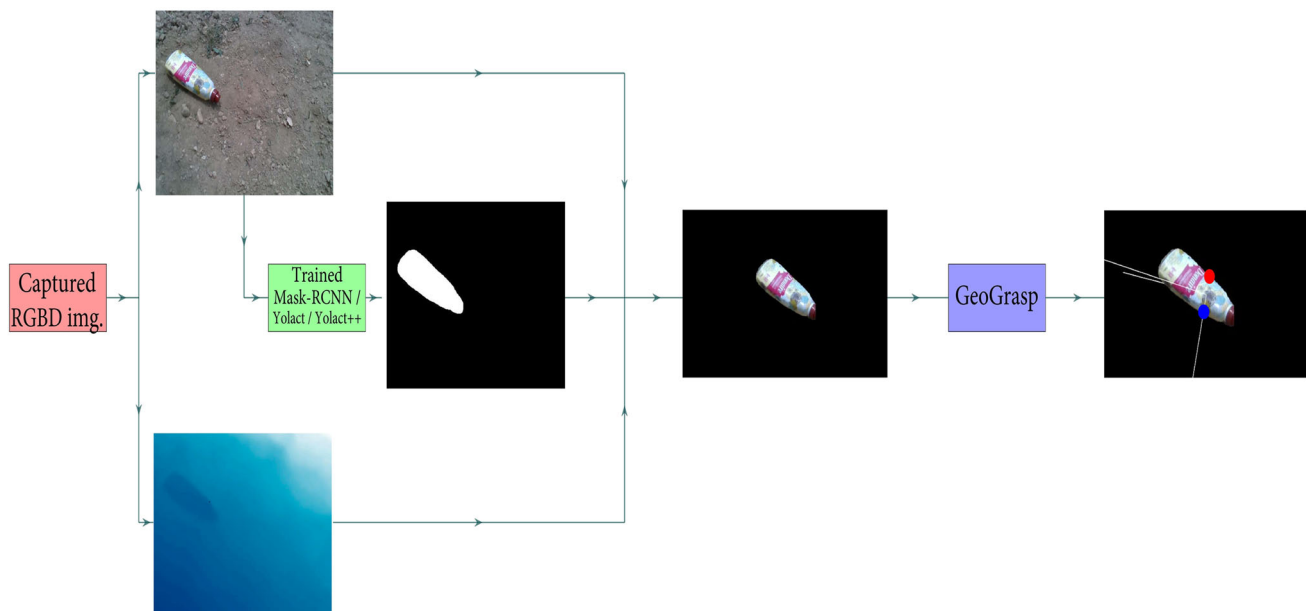
Fig. 5 Scheme of our object detection and grasping points calculation process. Our NNs obtain the segmented mask from the RGB image, which is used to extract the object point cloud in order to calculate the grasping points with our new version of the GeoGrasp algorithm

and some of them have, therefore, been prerecorded. These are from the navigation pose (see Fig. 1-bottom-left) to the home pose (see Fig. 1-left), the home pose to the detection pose (see Fig. 1-right) and the home pose to each of the container poses, and vice versa. In necessary, we simply need to play them. That will allow the robot to save computing time and avoid undesired or dangerous movements. The aforementioned trajectories are shown in Fig. 1-left. The remaining movements are planned in situ (blue paths - see Fig. 1-right). The green working area has dimensions of 600x500 mm. All objects in this area are easily graspable and easy for the camera to see.

Four containers are attached to the base of BLUE. This is the same as the number of categories used to classify objects in Sect. 3.1. Each class of the dataset will, therefore, be stored in different containers.

Finally, all positions have pre-positions. These are used to avoid pushing, moving, or colliding with the objects. There is a downward movement of ≈ 100 mm from the pre-position to the final position. This value was obtained empirically from the experiments and was that which obtained the best results.

3.3 Tactile manipulation

This module consists of detecting physical contact and slippage between the items of litter and the gripper during the manipulation task. This task involves picking up the item of litter from the ground and placing it in the desired bin, given the previously calculated trajectory and grasping points. In order to ensure grasping safety, the contact is detected before the robot lifts the object. Once the contact has been guaran-

teed, the robot will be able to adjust the grasping opening if a slip is detected. These operations cannot be carried out without tactile feedback in real time. We, therefore, implement a closed-loop controller for each task as shown in the blue scheme in Fig. 2.

It is known in the literature that force sensors are suitable for this kind of task because they allow the implementation of force-feedback controllers. Nonetheless, in this paper, the objective is to demonstrate that grasping and slipping controllers can be implemented in order to successfully solve the task by using optical tactile sensors that do not provide force values, which are known as DIGITs. These tactile sensors are more economical, smaller and provide more information about the features of the object such as texture or shape.

The DIGIT sensors provide tactile images and are used in order to implement the tactile feedback and closed-loop controllers. These sensors, which were originally designed in [9], contain a physical structure that is made up of: a reflective elastomer, an acrylic window, a 3D printed housing, a LED Printed Circuit Board (PCB) and a camera PCB. The sensor operates by recording the change in colours as a result of the deformation of the elastomer during the contact state. DIGIT sensors capture up to 30 Frames per Second (FPS) of RGB images with a resolution of 240x320 pixels. We mounted one DIGIT sensor (see Fig. 6) on each fingertip of the 2F-140 ROBOTIQ gripper. An additional 3D printed piece was designed and built so as to attach each sensor to each fingertip.

As the main contribution of this paper, we use touch images obtained from DIGIT sensors associated to contact properties to perform a tactile control. We design controllers to regulate the contact of a gripper to interact with objects

Fig. 6 (a) DIGIT sensor (b) Two DIGIT sensors mounted on a 2F-140 ROBOTIQ gripper

(a)

(b)
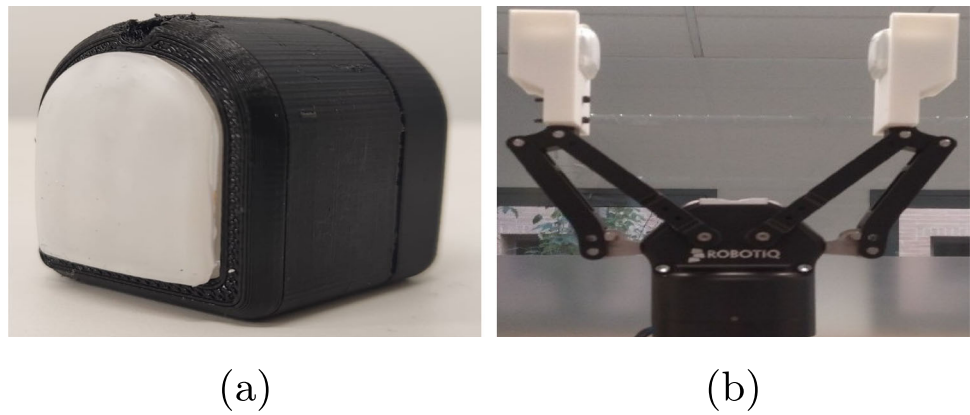
without mapping the force applied to the sensing surface. The force values cannot be reconstructed because this low-cost image-based sensor does not provide a pixel-to-force mapping as other optical sensors [36] or capacitive/resistive sensors [37]. The novelty consists in constructing a version of a tactile controller for unknown environments without using the tactile Jacobian and the features of the forces as in other works as [38].

3.3.1 Contact detection

Contact detection is formulated as a binary classification task as described in Algorithm 1. Given an input image $\varphi = [R, G, B]$ acquired from the DIGIT sensor, a ground truth label y ∈ [0, 1] is assigned to no contact and contact images (see Fig. 7), respectively. Bearing this in mind, we use a CNN to solve this task. This method was chosen owing to its ability to learn and extract features from images, such as edges and simple textures, to more complex textures, patterns and parts of objects.

In our previous work [39], we carried out extensive and rigorous experimentation with CNN architectures in order to discover which would be the most suitable for the purpose of *contact_prediction*. We trained three different architectures: VGG16 [40], InceptionV3 [41] and MobileNetV2 [42] following a transfer learning strategy. Our dataset contained ≈ 16.000 images, which were manually annotated by comparing a contact image with a no-contact image reference, from three DIGIT units and nine objects with different shapes and textures. This work showed that InceptionV3 was the most appropriate architecture in terms of accuracy, robustness, and inference time. However, in the present work, it is necessary to reduce the size of its architecture in order to speed up the inference process in our embedded system for the robot. This reduction led to the decision to train one NN ($\theta_{\text{contact}}^{\text{sensor\_unit}}$) can be unit A or B, thus, improving performance and evaluation values. Specifically, we used the InceptionV3 backbone up to the "mixed5" layer as a feature extractor and modified the final layers to adapt them to our task. The final layers are made up of a GlobalAveragePooling2D layer and

two blocks of Batch Normalisation, Dense, and Dropout layers. Finally, the output layer was added with a single neuron and a sigmoid activation function with a threshold $T_{\text{contact}}^{\text{sensor}}$ (Fig. 8).

In order to execute the desired manipulation tasks, a closed-loop controller is implemented, as shown in Algorithm 1. First, it checks whether the pose of the robot is that of grasping or releasing the item of litter. It then executes the contact prediction model to obtain the contact state for each sensor. When carrying out the subsequent grasping task, the robot closes the gripper by one step each time that the *contact_prediction* is equal to 0, signifying that the item of litter has not yet been grasped. During the releasing task, however, the robot opens the gripper by one step each time that the *contact_prediction* is equal to 1, signifying that the item of litter has not yet been released. The execution of Algorithm 1. ends when the *contact_prediction* is equal to 1 (grasping task) or 0 (release task).

Algorithm 1 Grasping-Contact algorithm

1: **repeat**
2:   **if** *robot_in_pose* = 1 **then**   ▷ Grasp or release pose
3:     norm_image_sensorA←*normaliseInput*(*image_sensor A*)
4:     norm_image_sensorB←*normaliseInput*(*image_sensor B*)
5:     contact_prediction_A ← $\theta_{\text{contact}}^{\text{sensorA}}$(*norm_image_sensor A*)
6:     contact_prediction_B ← $\theta_{\text{contact}}^{\text{sensorB}}$(*norm_image_sensor B*)
7:     contact_prediction   ←   *contact_prediction_A* ∧ *contact_prediction_B*
8:     **if** *task* = *grasp* **then**   ▷ Grasp litter object
9:       **if** *contact_prediction* = 0 **then**   ▷ Detected no contact
10:          closeGripper(1)   ▷ Closing one step of the motor
11:       **else**   ▷ Detected contact
12:          done ← 1   ▷ End
13:       **end if**
14:     **end if**
15:     **if** *task* = *release* **then**   ▷ Release litter object
16:       **if** *contact_prediction* = 1 **then**   ▷ Detected contact
17:          openGripper(1)   ▷ Opening one step of the motor
18:       **else**   ▷ Detected no contact
19:          done ← 1   ▷ End
20:       **end if**
21:     **end if**
22:   **end if**
23: **until** *done* = 1

(a) Unit A   (b) Unit A   (c) Unit B   (d) Unit B
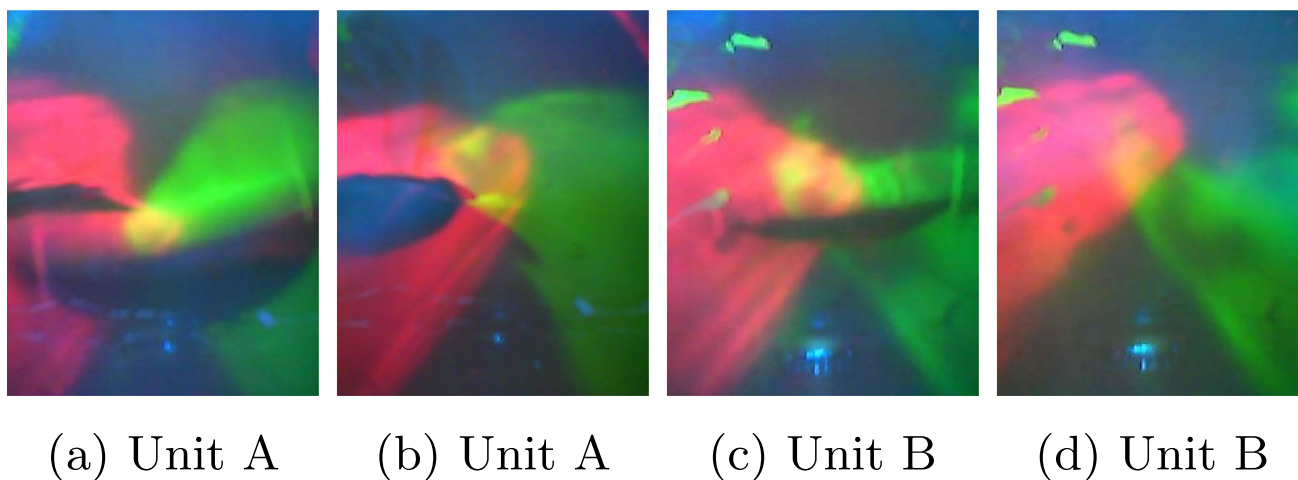
Fig. 7 Contact (a,c) and no contact (b,d) *image_sensor* from DIGIT units A (a,b) and B (c,d). The images from both sensors are not identical because both sensors were manufactured by hand

3.3.2 Slip detection

Various approaches with which to solve the slip detection task by classifying image sequences have been proposed in the literature (see Sect. 2.2). In contrast, this paper presents a grasping method based on slip detection as described in Algorithms 2 and 3. Our proposal formulates the stage of slip detection as an image classification problem between two classes: slip and stable. The slip class refers to object movement during robot manipulation, while the stable class implies no movement.

The slip detection (Algorithm 2) takes as input a grayscale image sequence $\Phi = [\delta_t, \delta_{t+1}, \delta_{t+2}, \delta_{t+3}]$, whose length is empirically calculated in order to attain the best results in terms of accuracy and inference time and where $\delta = 0.299 \times R + 0.587 \times G + 0.114 \times B$. A subtraction oper-ation $\psi = \delta_{t+3} - \delta_t$ is applied to obtain the changes in the deformation of the elastomer. The subtracted image ψ is very noisy because the pixel values are not identical in consecu-tive images. ψ is, therefore, filtered by applying an opening morphological operation $\Psi = \psi \circ \kappa = (\psi \ominus \kappa) \oplus \kappa$, where $\circ, \ominus, \oplus$, and κ denote opening, erosion, dilation and a struc-turing element, respectively. Ψ is a binary image (black and white) that represents two possible states or labels $y \in [0, 1]$. Label $y = 0$ is assigned to the stable class and label $y = 1$ is assigned to the slip class. Figure 9 shows examples of this pre-processing in which slip images produce white patterns (pixel value of 255), and stable images are almost black (pixel value of 0). The subtraction ψ and filtering Ψ operations cor-respond to the *filterImage* function in Algorithm 2.

Once Ψ has been obtained, it can be classified as apper-taining to the slip or stable class by using two different

Input image
[320,240,3]

InceptionV3 architecture
until "mixed5" layer

Global Average
Pooling Layer

BatchNorm,
Dense and
Dropout block 1

BatchNorm,
Dense and
Dropout block 2
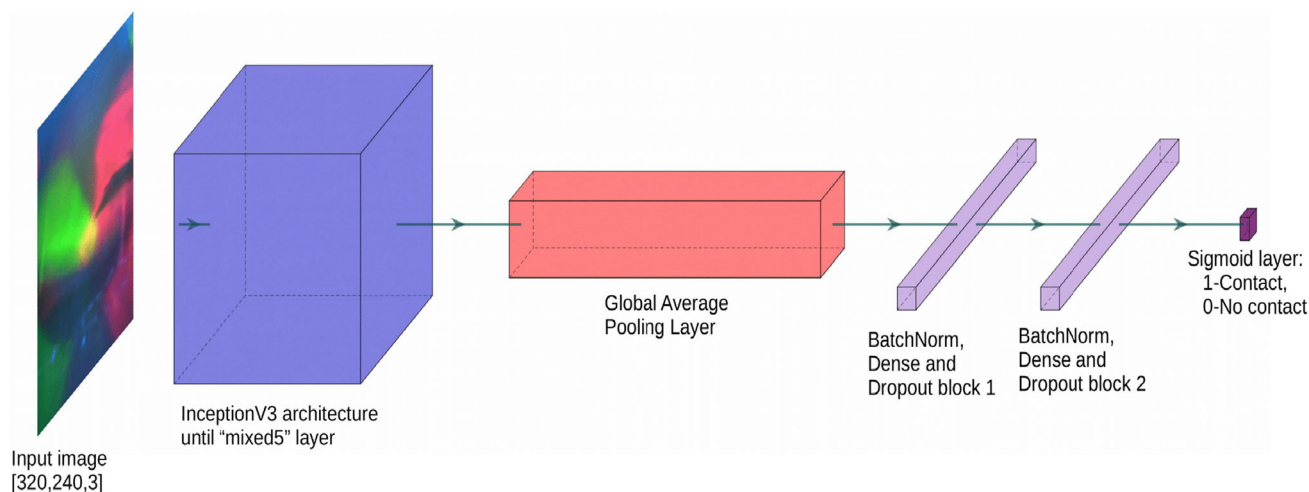
Sigmoid layer:
1-Contact,
0-No contact

Fig. 8 Architecture of our $\theta_{\text{contact}}^{\text{sensor\_unit}}$. It is based on the well-known InceptionV3 architecture, although the final layers have been modified
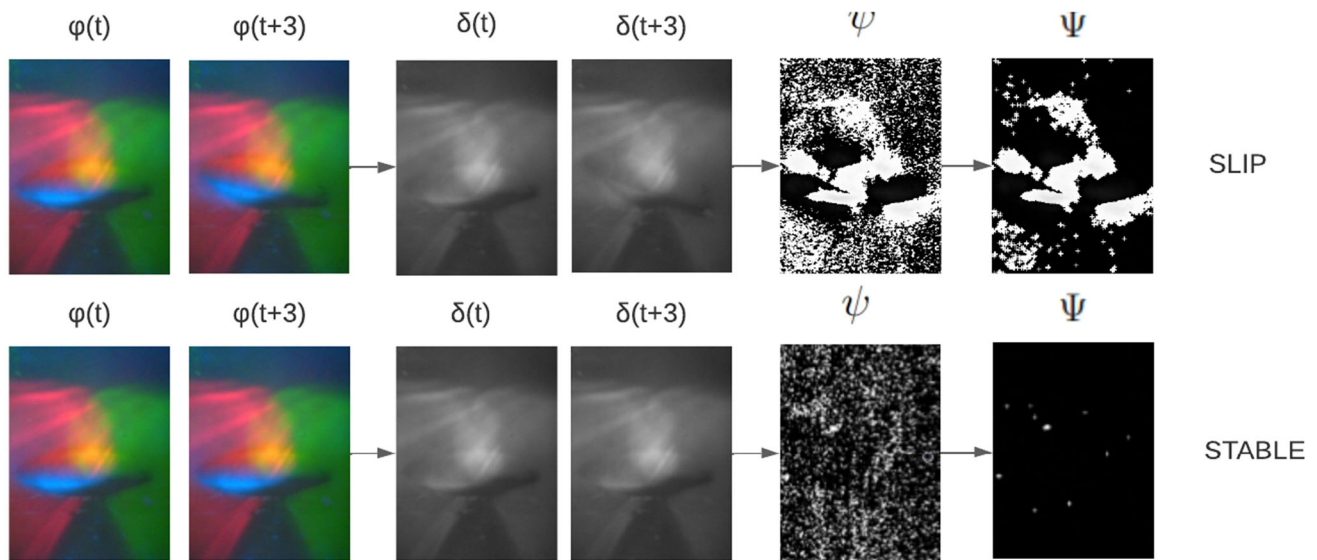
Fig. 9 Example of pre-processing steps employed to obtain the final image before the classifying process

approaches and thresholds, which are compared and justified in Sect. 4.4. The first consists of calculating the brightness of the Ψ image, in which a threshold value of $T_{\text{slip}}^{\text{brightness}}$ is established as the final classifier. The second method is a CNN ($CNN\,Prediction$ in Algorithm 2) whose architecture is described in Fig. 10, followed by a threshold value $T_{\text{slip}}^{\text{cnn}}$ as the final classifier. Detailed descriptions of both methods are provided in Algorithms 2 and 3. The robot later closes the gripper by one step each time that the slippage event occurs. The execution ends when the robot is in the pose required to release the object.

4 Experimental results

4.1 Setup, performance metrics, and data

This section provides descriptions of first the hardware devices used to train and test the NN, second the performance metrics used to express our results, and finally, general information regarding our datasets.

The first device is an NVIDIA A100 Tensor Core GPU with 40 GB of RAM memory. This device was used to train the visual and tactile perception modules. The other is an NVIDIA Jetson AGX Xavier board. This device was used to test both modules and for real-time execution.

With regard to the evaluation metrics used, we have tested our visual system with an AP metric [43], [44] and our tactile system with an accuracy metric [45]. These metrics are well-established in literature and expressed our results in a complete and reliable manner.

Algorithm 2 Slip-Detection algorithm

1: **function** $slipDetection(image\_sensor\_sequence, method)$
2:    $\Psi \leftarrow filterImage(image\_sensor\_sequence)$
3:    **if** $method = brightness$ **then**
4:       brightness_value $\leftarrow \frac{\sum_{i=0}^{h-1}\sum_{j=0}^{w-1}\Psi(i,j)}{h \times w}$ ▷ (h, w): Ψ height and width
5:       **if** $brightness\_value \geq T_{\text{slip}}^{\text{method}}$ **then**
6:          grasping_state $\leftarrow 1$
7:       **else**
8:          grasping_state $\leftarrow 0$
9:       **end if**
10:    **else**
11:       grasping_state $\leftarrow CNN\,Prediction(\Psi)$
12:       **if** $grasping\_state \geq T_{\text{slip}}^{\text{method}}$ **then**
13:          grasping_state $\leftarrow 1$
14:       **else**
15:          grasping_state $\leftarrow 0$
16:       **end if**
17:    **end if**
18:    **return** grasping_state
19: **end function**

Algorithm 3 Grasping-Slip algorithm

1: **repeat**
2:    **if** $object\_is\_grasped = 1$ **then**
3:       grasping_state_A $\leftarrow slipDetection(image\_sensor A\_sequence, method)$
4:       grasping_state_B $\leftarrow slipDetection(image\_sensor B\_sequence, method)$
5:       final_grasping_state $\leftarrow$ grasping_state_A $\vee$ grasping_state_B
6:       **if** $final\_grasping\_state = 1$ **then** ▷ Detected slip
7:          closeGripper(1) ▷ Close to avoid object falling
8:       **end if**
9:       robot_in_release_pose $\leftarrow checkRobotInReleasePose()$
10:    **end if**
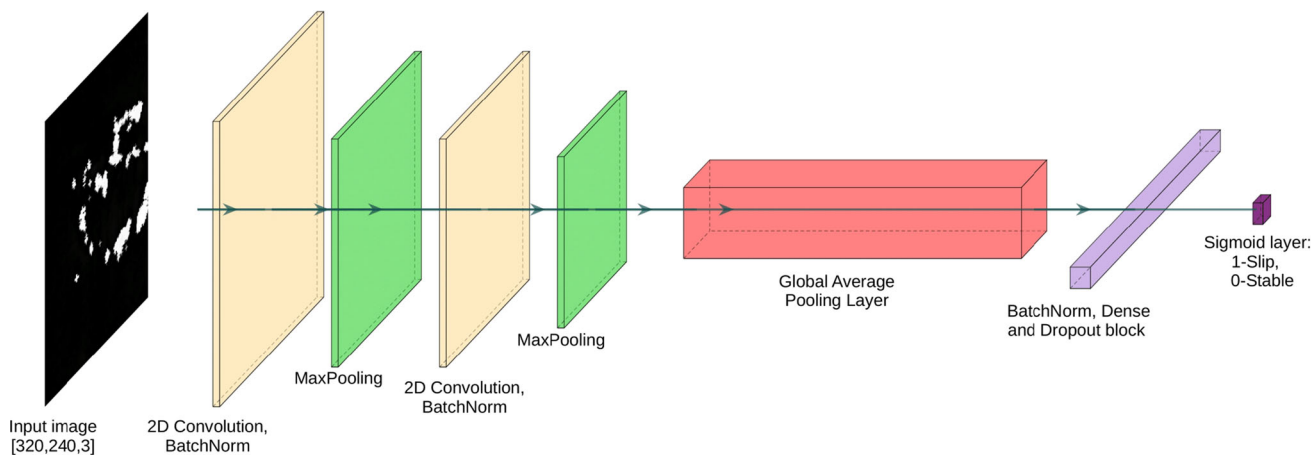11: **until** $robot\_in\_release\_pose = 1$

Fig. 10 Architecture of $CNN\,Prediction$, which is based on a simple CNN architecture for image classification

The AP metric, which is described in Eq. (3), is calculated for different IoU thresholds, described in Eq. (4).

$$AP_{\text{IoU}} = \sum_{i \in \tilde{r}} (r_{i+1} - r_{i}) \cdot \max_{\tilde{r}:\tilde{r} \geq r_{i+1}} \rho(\tilde{r}) \qquad (3)$$

$$IoU = \frac{area(gt \cap pd)}{area(gt \cup pd)} \qquad (4)$$

where $r, \rho, \tilde{r}, gt$, and pd denote levels of recall, precision, and recall values, the ground truth, and the prediction bounding boxes. All these values are calculated to measure the performance of the NN models of our pipelines.

The accuracy metric described in Eq. (5) can be used for tactile testing because the tactile datasets are well-balanced.

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \qquad (5)$$

where TP, TN, FP, and $FN \in \mathbb{N}_{\geq 0}$ and denote True Positives, True Negatives, False Positives, and False Negatives. A TP detection means that the system detects contact or slip and this is correct, while a TN detection means that the system detects no contact or no slip and this is also correct. FP and FN detections occur when the system detects that there is contact or slip but this is not correct or when the system does not detect a contact or slip state but it exists, respectively.

We created three datasets for each specific task that required a training phase: vision-based waste detection (D1), tactile-based contact detection (D2), and tactile-based slip detection (D3). For the visual module, we used 52 different household objects for the four classes (plastic, cardboard, glass, and metal), while for the tactile tasks we used only eight and six objects, for contact and slip detection, respectively. The visual module requires more objects for the training phase in order to learn a large variety of shapes, colours, etc., in different outdoor scenarios. Moreover, with regard to the tactile manipulation modules, different objects may produce

similar tactile images because they share similar shapes and geometries, signifying that the tactile datasets do not require such a high number of objects.

Table 1 shows that the D2 dataset contains more samples because it is easier to collect and label tactile images than household waste images in different environments. The D3 samples are also tactile images, but as explained earlier, a sequence of four gray-scale images is transformed into a single binary image in order to detect slip, and the final number of images is, therefore, smaller than in the rest.

4.2 Data collection and training for visual perception

This section describes the training dataset, the training phase, and the results both with validation and test sets.

As there are not many household waste datasets containing objects that are dirty or partially destroyed, we had to create one ourselves (dataset D1). Images taken in this kind of environment were used for the training task. These environments are all at the Technological Scientific Park, in the area around our university, and include asphalt, pebbles, and green backgrounds. The objects in them are closer and further away, in addition to being partially occluded or shadowed by other elements in the scene and having different lighting conditions (see Fig. 11).

Our system learns to classify and instantiate household waste into four different categories: plastic, cardboard, glass, and metal.

Table 1 Number of total samples of each dataset for all classes, sets, and sensors

	D1	D2	D3
Number of objects	52	8	6
Number of total samples	6943	24607	3540

Fig. 11 Variability environments in our dataset. They show different visual and physical features

Table 2 Distribution of dataset D1

	Plastic	Cardboard	Glass	Metal
Images	1776	1626	1769	1772
Different objects	21	9	10	12
Percentage	25.58 %	23.42 %	25.48 %	25.52 %

Table 3 Results of validation dataset using AP

Algorithm	Backbone	AP_{50}	AP_{75}	AP_{90}
Mask R-CNN	ResNet50	0.966	0.934	0.595
Mask R-CNN	ResNet101	0.971	0.945	0.614
YOLACT	ResNet50	0.994	0.982	0.690
YOLACT	ResNet101	0.995	0.979	0.637
YOLACT	DarkNet53	0.997	0.983	0.720
YOLACT++	ResNet50	0.998	0.985	0.765
YOLACT++	ResNet101	0.998	0.980	0.754

Table 4 Results of test dataset using AP and inference time. The inference time is calculated as the mean average of every image detected in the test set

Algorithm	Backbone	AP_{50}	AP_{75}	AP_{90}	Inference time (ms)
Mask R-CNN	ResNet50	0.958	0.935	0.585	74.835
Mask R-CNN	ResNet101	0.967	0.945	0.612	86.032
YOLACT	ResNet50	0.998	0.975	0.701	18.187
YOLACT	ResNet101	0.996	0.968	0.661	27.644
YOLACT	DarkNet53	0.998	0.980	0.696	17.213
YOLACT++	ResNet50	0.994	0.975	0.731	21.085
YOLACT++	ResNet101	0.999	0.974	0.747	29.831

Table 5 Results of test dataset using AP as metrics. These results are calculated per object class and using YOLACT NN with DarkNet53 as a backbone

	Plastic	Cardboard	Glass	Metal	Total
AP_{50}	1.000	1.000	0.996	0.996	0.998
AP_{75}	0.989	0.988	0.972	0.971	0.980
AP_{95}	0.800	0.771	0.582	0.631	0.696

Fig. 12 Examples of outdoor litter detection. These examples contain one object from each class (cardboard, plastic, metal, and glass) in different environments

In fact, each class represents a broad range of elements made of each material (see Table 2). There are differently posed and sized water bottles, drinking bottles used by practitioners of sports, Tupperware containers, cans, glass beer bottles, or juice cardboard, among others. This way of naming the classes is common to other datasets and will facilitate the addition of new images from other sources if necessary. These images were extracted from video files. The video was recorded at a resolution of 640x480 pixels, using a RealSense™ D435i depth camera. As there is only one object per image, the number of images per class and per instance coincide. After processing the videos, 6,943 images were obtained to compose the dataset D1. These images were labeled with LabelMe [46], an image annotation tool. As will be noted, all the classes are well balanced (see Table 2).

For our experimentation, we split the dataset D1 into training, validation, and test sets. This was done by following the 70/20/10 proportion. The division is made randomly to guarantee that results are not dependent on how data is distributed and picked. Indeed, objects in the test set have not been used during training or validation phases before.

The learning process of the NNs has the following methodology. For Mask R-CNN and all its derivates, the training was split into three sub-training tasks. The first consisted of training the network heads for 40 epochs. The second lasted 80 epochs and trained ResNet backbone stage four and upwards. The last involved training the full NN for 40 more epochs. In the first two sub-stages, the learning rate did not change, while in the last one, it was reduced to 10 times its original value. We used 2 images per GPU and 1,000 steps per epoch, signifying that a random subset of 2,000 images was used for training in each epoch. The learning rate was set to 0.001 using the Stochastic Gradient Descent (SGD) optimizer.

Only one training task was applied to both the second and the third set. This consisted of training the complete NN for 160 epochs. The learning rate was reduced by 2e-7 during the first two and a half epochs, from 0.0011 to 0.0010. There were also 26 images per GPU and 216 steps per epoch, signifying that each image in the training dataset was used per epoch. With regard to the optimizer, SGD was again used. The following results were obtained after the training process had been carried out (see Tables 3 and 4). These results are expressed using AP_{50}, AP_{75}, and AP_{90} as metrics (AP values between 0 and 1).

During the training process, YOLACT++ with ResNet50 obtained the best results with the validation dataset. It achieved 99.8%, 98.5% and 76.5% in AP_{50}, AP_{75} and AP_{90} respectively. The first two metrics did not help much as regards choosing an algorithm, but the last allowed us to choose YOLACT as the best method. After using the test dataset, the final results were obtained and they had a similar trend. The best model was again YOLACT++ with ResNet101 in both AP_{50} and AP_{90}, achieving a score of 99.9% and 74.7% respectively, while in AP_{75} the best model
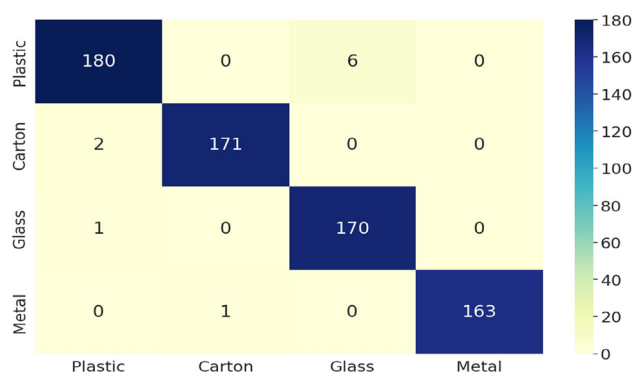

Fig. 13 Test confusion matrix with the NN selected. Rows indicate true labels and columns indicate predicted labels

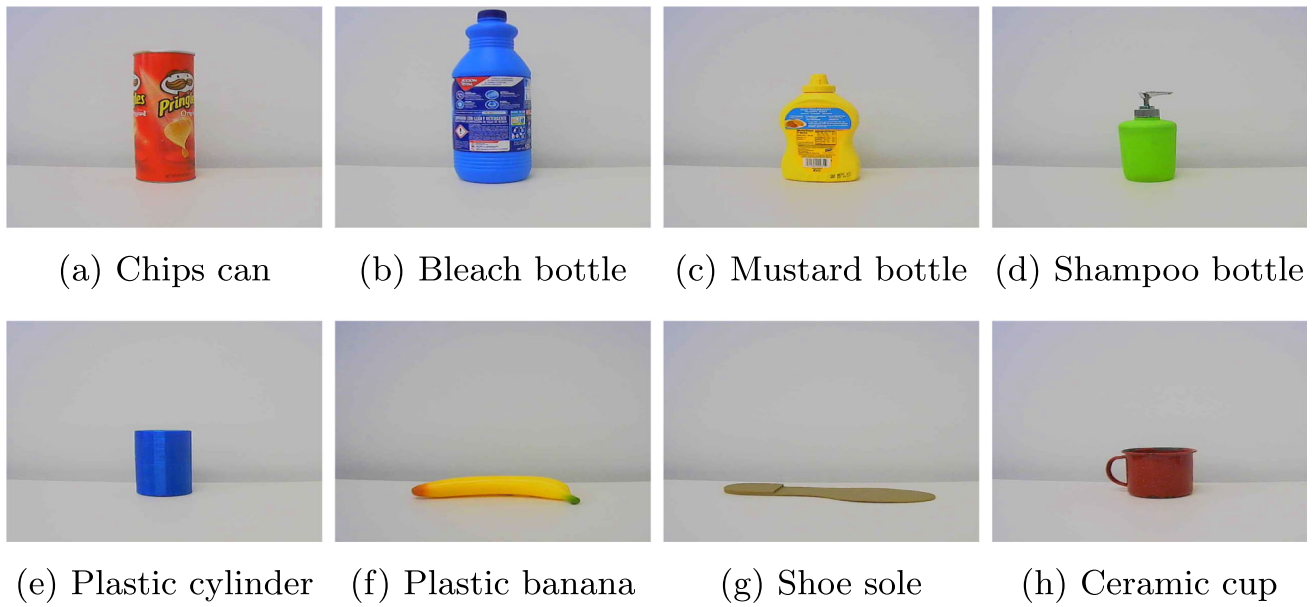| (a) Chips can	(b) Bleach bottle	(c) Mustard bottle	(d) Shampoo bottle
(e) Plastic cylinder	(f) Plastic banana	(g) Shoe sole	(h) Ceramic cup

Fig. 14 Objects used to create the dataset D2, which was used to train, validate and test the contact detection method and also to train the slip detection CNN

Table 6 Distribution of images per sensor unit $[D2_A, D2_B]$ considering the class (contact, no contact) in training, validation, and test sets

	Train		**Validation**		**Test**	
Class	A	B	A	B	A	B
Contact	4035	4683	1224	1344	603	663
No contact	3803	4683	1092	1345	539	663

Table 7 Accuracy (Acc) values for sensors A and B, and $T_{\text{contact}}^{\text{sensor\_unit}} = 0.4, 0.5, 0.6$. Accuracy values are between 0 and 1

Sensor unit	Acc (T=0.4)	Acc (T=0.5)	Acc (T=0.6)
A	0.969	0.964	0.963
B	0.996	0.996	0.994

Fig. 15 Translational (red and green) and rotational (blue) perturbations produced by a human operator. These movements include all perturbations that the object could undergo during the manipulation task

Table 8 Distribution of images (Ψ) per sensor unit $[D3_A, D3_B]$ considering class (stable, slip) in training and validation sets

	Train		Validation	
Class	A	B	A	B
Stable	620	620	265	265
Slip	627	613	268	262

(a) Deformable toy   (b) Plastic toy   (c) Salt bottle
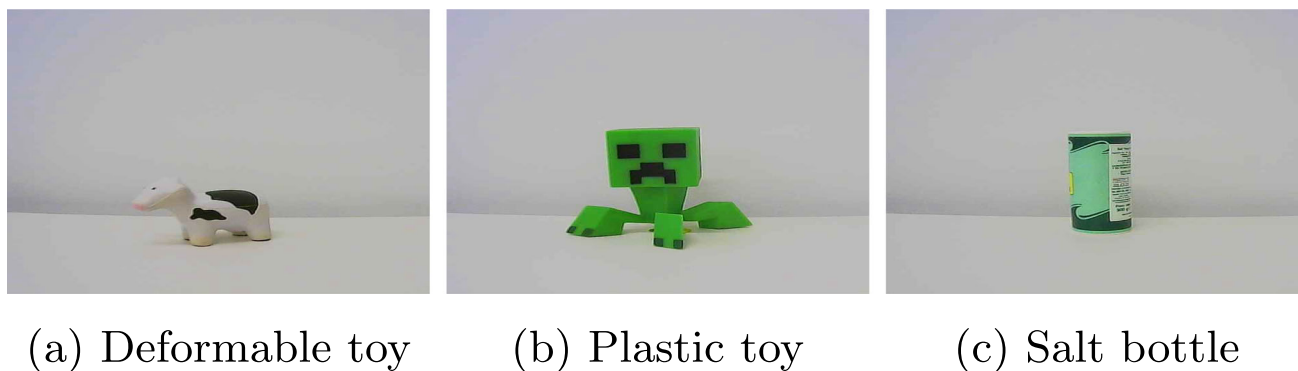
Fig. 16 Objects in our test set. These objects differ from those in the training set in terms of size, textures, deformability, etc

was YOLACT with DarkNet53, achieving an average precision of 98%. Since we have two different NNs with bigger and smaller backbones, inference time is also a useful tool for the selection of the best NN model. This will help to determine which combination is the fastest in the inference task. The results are shown in Table 4. As will be observed, the fastest model is the combination of YOLACT with DarkNet53, achieving 17.2 ms. There is a huge step between the YOLACT and Mask R-CNN models, with YOLACT being an average of more than 50 ms faster.

All of these results led to the decision that the NN chosen for the process would be YOLACT with DarkNet53 as a backbone. We chose this NN since we prioritise speed (it is 57% faster when compared to that with the best AP_{90}) over accuracy and also because AP_{75} is sufficiently good for our process. The accuracy of the NN per class in our dataset is shown in Table 5.

We also show some outdoor recognition examples when our visual model is used with unknown samples from the four classes. These are provided in Fig. 12. The following confusion matrix was obtained when using the NN chosen (see Fig. 13), in which the most confusion occurs between the plastic and glass classes.

4.3 Tactile data collection and training for contact detection

This section describes the tactile dataset D2 for the contact detection task, the training phase, and the results with the test set.

The tactile data was collected by performing consecutive maneuvers and recording the images from the DIGIT sensors mounted on the gripper. A human operator changed the pose of the objects for each robotic grip.

Eight objects were used to create the dataset D2 = $[\varphi_1, \varphi_2, \varphi_3, ..., \varphi_n]$ where n is the total number of images. Approximately the same number of images was obtained for each object. Eight objects were sufficient to form the dataset because they differed in terms of size, shape, deformation, weight, texture, and material (see Fig. 14). We discarded objects that are completely deformable or very narrow such as plastic bags or cardboard sheets due to the limitations of the fingertips and tactile sensing area in size.

We train our CNN (see Sect. 3.3.1) with each sensor, units A and B, because the images extracted from DIGIT sensors are not identical (see Fig. 7). We, therefore, achieved better results with individual models rather than a single model for all the sensors. Two datasets $[D2_A, D2_B]$ were, therefore, designed (see Table 6), and each dataset was split into three subsets: training (70%), validation (20%) and test (10%). The dataset is roughly balanced in terms of sensor and class samples. This procedure enabled our NNs to learn and check their knowledge with 90% of the data during the training phase. The remaining 10% of the data was used to perform an evaluation test once the training phase had been completed.

A transfer learning strategy was applied in order to carry out the training phase. The layers up to the 'mixed5' layer were, therefore, set as being non-trainable. The remaining layers were set as trainable, including the head. A Root Mean Squared Propagation (RMSProp) optimizer was used with a learning rate of 3×10^{-6}, a batch size of 24, and a binary cross entropy loss. Once both models had been trained, the evaluation process was executed on the test dataset. The results are expressed in terms of the accuracy metric, previously

Table 9 Results obtained using CNN method for sensors A and B, and T_{slip}^{cnn} = 0.4, 0.5, 0.6

Sensor unit	Acc (T=0.4)	Acc (T=0.5)	Acc (T=0.6)	Time(ms)
A	0.733	0.733	0.689	157.5
B	0.689	0.689	0.644	157.5

Table 10 Results obtained using brightness method for sensors A and B, and $T_{\text{slip}}^{\text{brightness}}$ = 5, 10, 15

Sensor unit	Acc (T=5)	Acc (T=10)	Acc (T=15)	Time(ms)
A	1	1	0.911	7.5
B	1	1	0.911	7.5

described in Eq. (5). Table 7 shows accuracy values for both the sensors and the three different thresholds ($T_{\text{contact}}^{\text{sensor\_unit}}$). Although these values are very similar, $T_{\text{contact}}^{\text{sensor\_unit}} = 0.5$ was chosen in order to prevent wrong detections resulting from the hysteresis of DIGIT sensors. This hysteresis is produced when the elastomer is recovering its initial shape after the contact.

4.4 Tactile data collection and training for slip detection

This section describes the tactile dataset D3 for the slip detection task, the training or tuning phase, and the results with the test set.

$D3 = [\Psi_1, \Psi_2, \Psi_3, ..., \Psi_n]$ where n is the total number of images. As slip detection is a different task to contact detection, it was necessary to create this new dataset in order to capture the corresponding tactile images generated when the objects slip. The slip class images were generated by applying three external instabilities to the object, while the stable class images were regular images with no disturbances. As noted in Fig. 15, one rotational and two translational movements were applied to each object. The goal was to detect not only slippage, but also any other possible perturbations during the robot manipulation. These perturbations could result in the object falling to the ground, thus, generating a collection failure.

Six objects (b, c, e, f, g, and h in Fig. 14) were selected from the set of eight used for tactile detection. They were sufficiently varied to be able to form the training dataset, which was divided into two subsets: training (70%) and validation (30%) (see Table 8). Objects a and d were discarded from D3 in order to avoid repeating cylindrical shapes. The number of images obtained for the six objects was roughly balanced and the images were different from those in the contact detection dataset.

Three novel objects were also added to D3 for testing (see Fig. 16). In this way, we can evaluate the generalisation capabilities of the proposed algorithms with previously unseen objects.

No training process was required in order to calculate the brightness of an image. The CNN did, however, require this training process. The low number of parameters allowed us to train the full network from scratch. An Adam optimizer [47] was used with a learning rate of 1×10^{-4}, a batch size of 32, and a binary cross entropy loss.

After the training process had been completed, the performance of the model was calculated with the test set. This set was different from the training and validation sets because it was not made up of shuffled images with their respective labels. The test set contained three continuous videos, one per object, to which we applied these three external instabilities, five times each. This signifies that there are 15 instabilities per object, and 45 in total per sensor.

The aim of this experiment was for the models to detect every case of instability as a slip class without detecting false positives. The results are expressed in terms of instability-detection accuracy and inference time in our embedded system. The accuracy values are between 0 and 1 and the total time in each timestamp is calculated by adding the inference time of each sensor.

Tables 9 and 10 show that detecting slippage by calculating the brightness of Ψ images is more accurate and faster than using a CNN to classify these type of images. This is caused because the $filterImage$ function (from Algorithm 2) converts the RGB tactile images into binary images, from which the CNN has more difficulties to extract features.

The $T_{\text{slip}}^{\text{brightness}}$ threshold values of five and ten achieve the same accuracy value, but with $T_{\text{slip}}^{\text{brightness}} = 5$, the model frequently detects false positives. We consequently set the brightness method with $T_{\text{slip}}^{\text{brightness}} = 10$ as the slip detector.

4.5 Detection and manipulation results in outdoor

In this section, we show the experimentation that we carried out in order to test our system in three different outdoor environments with different objects with respect to the previous sections. Finally, we show the promising and reliable results
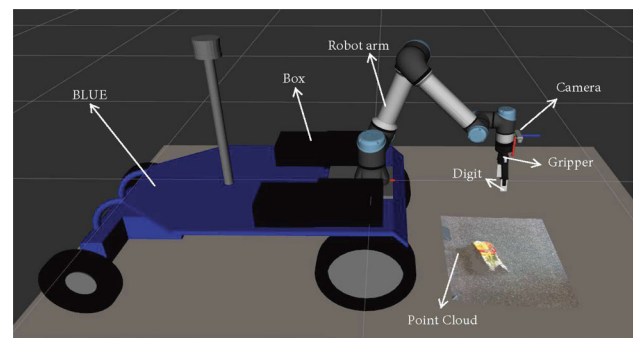

Fig. 17 Robotic system that allows tasks to be performed in simulated environments
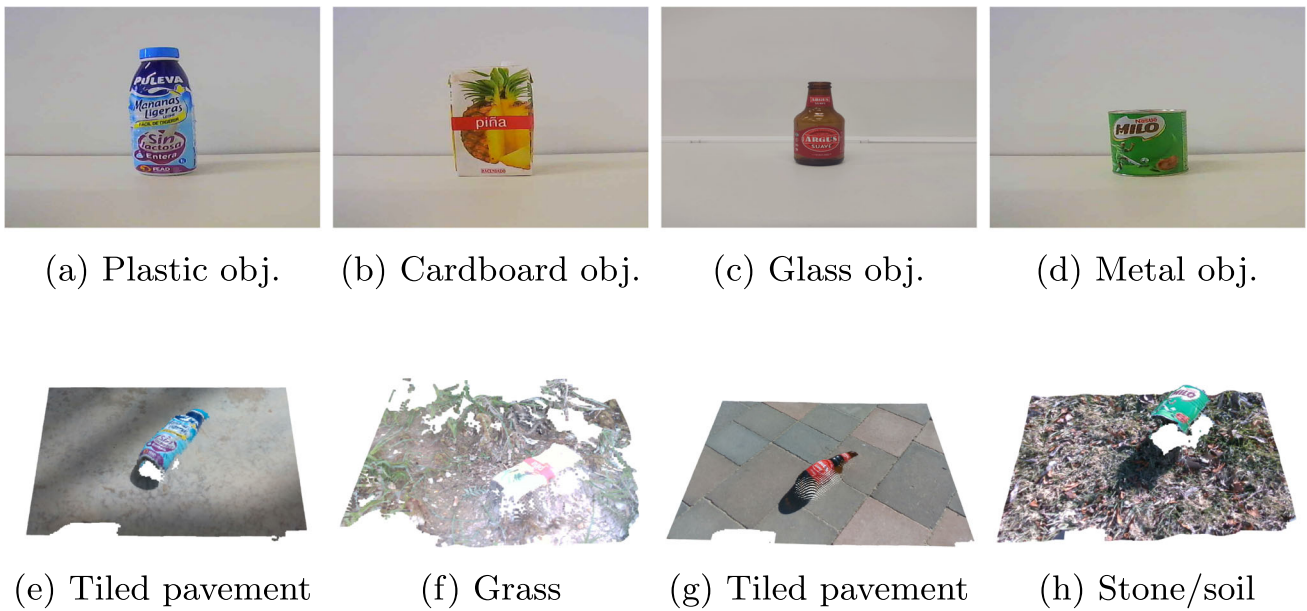
(a) Plastic obj.   (b) Cardboard obj.   (c) Glass obj.   (d) Metal obj.

(e) Tiled pavement   (f) Grass   (g) Tiled pavement   (h) Stone/soil

Fig. 18 Objects (one from each class) and environments (3 different environments) used to test our robot in real outdoor experiments
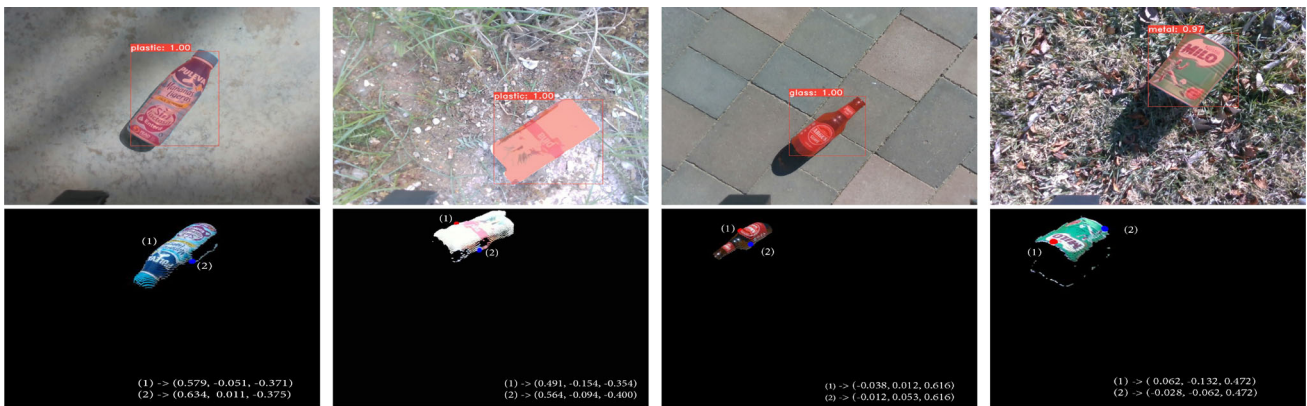

Fig. 19 Examples of YOLACT detection (top row) and grasping points found by GeoGrasp using robot (global) frame (bottom row)
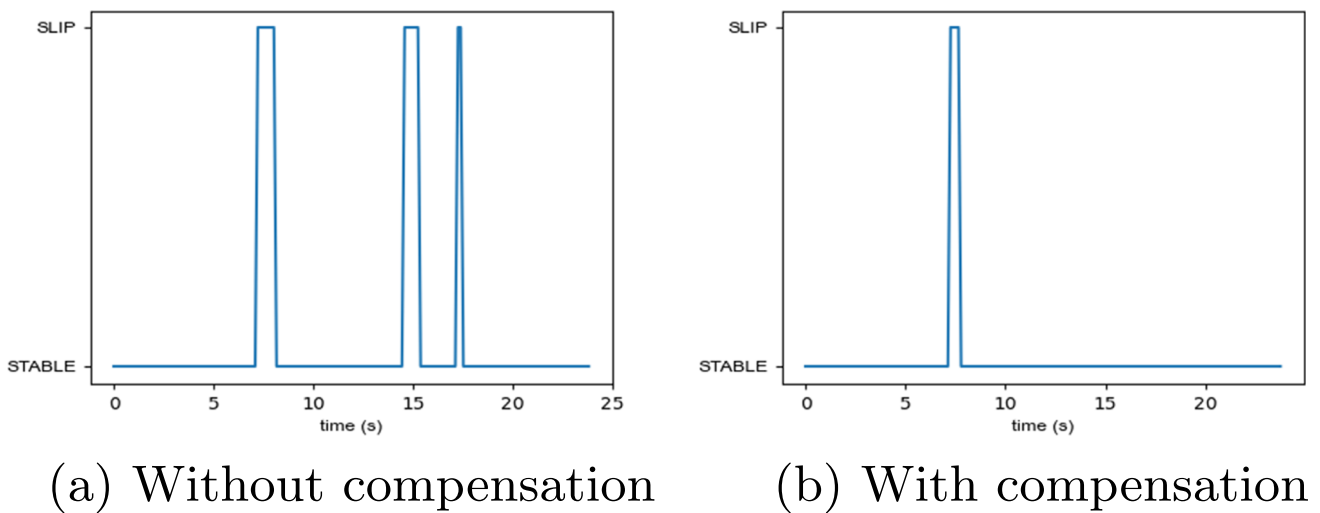

(a) Without compensation   (b) With compensation

Fig. 20 Number of slip detections while the robot arm is lifting the object. This number is smaller when applying grasping compensation

obtained from this experimentation which prove that our system is able to perform the litter collection task. However, prior to running certain tasks in real mode, it is recommendable to simulate them in similar conditions. We specifically simulated the trajectory planning so as to visualise and check the robot arm and the gripper movement that is necessary in order to accomplish the robotic grasping task. We did this by building our mobile manipulation robot using ROS and RVIZ. Our simulation comprised the BLUE robot, the UR5e robot arm with the gripper and the RGBD camera mounted on the end effector, along with DIGIT sensors mounted on the fingertips of the gripper, as shown in Fig. 17.

To complete the outdoor experiment, we decided to use four new objects, one from each of the classes: cardboard, plastic, metal, or glass. These objects had not been used to form any of the previous visual or tactile datasets and were employed with the objective of testing the generalisation capabilities of our system with unknown objects for the waste collection task. Testing our system in different environments brought this experiment closer to reality. We, therefore, chose three new environments that were not been included in our previous datasets in order to test the generalisation of our proposal in unforeseen situations (see Fig. 18).

The following images show the results obtained for the YOLACT detection and grasping points (Sect. 3.2). Once the coordinates of these points have been referenced to the camera, it is necessary to obtain their global position. This position will be obtained from the robot base. Some of these coordinates are shown in Fig. 19.

Before carrying out the main outdoor experiment, two sub-experiments were carried out in order to establish the variables of the tactile manipulation module. The goal of the first sub-experiment was to demonstrate that sending a closing command through ROS to the gripper when slippage was detected would make the grasping more stable. In the second, we established a variable to denote the number of contact detections predicted by the model, in order to consider that the object had been grasped. Finally, in the main outdoor experiment, we evaluated our system by running our pipeline five times per object in each environment, namely, 60 object pick ups.

In order to demonstrate that grasping required compensation, we filled one object with a small quantity of water. The idea was to perform the grasping task, lift the object and evaluate the slip detections with and without grasping compensation. Figures 20 and 21 show that when grasping
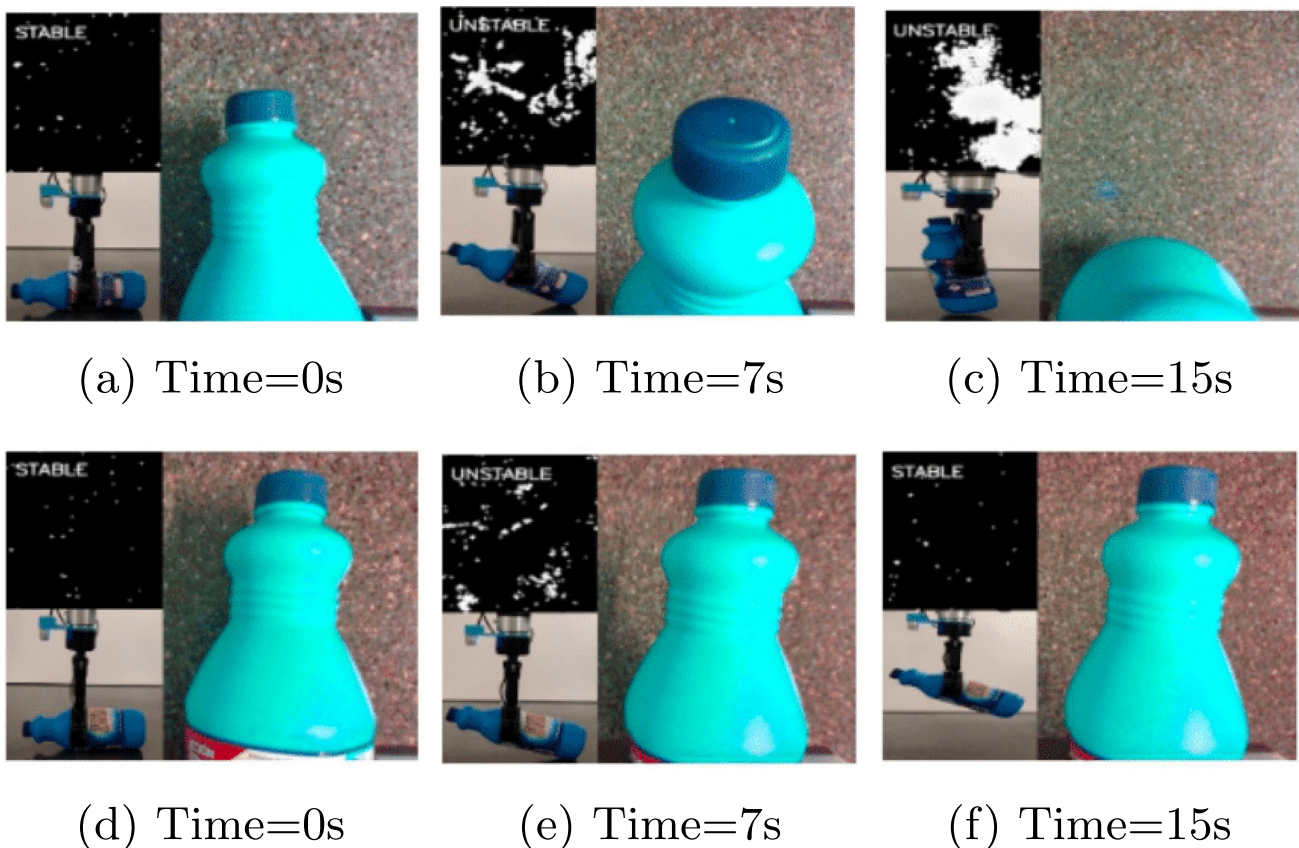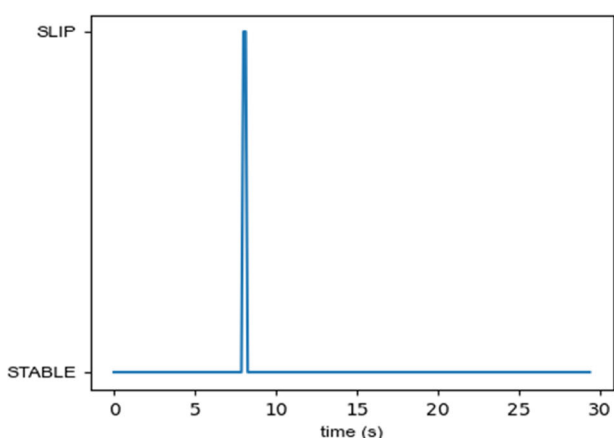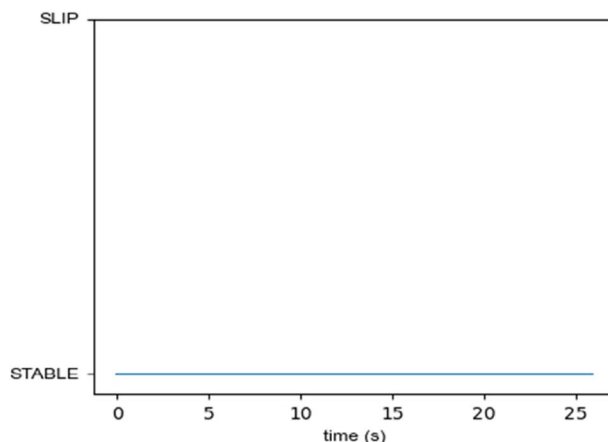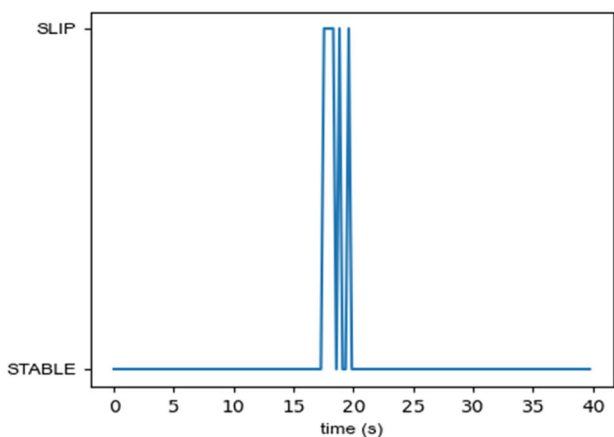

(a) Time=0s   (b) Time=7s   (c) Time=15s

(d) Time=0s   (e) Time=7s   (f) Time=15s

Fig. 21 (a, b, c) Object falling without grasping compensation. Slip state is produced and detected (unstable). (d, e, f) Object does not fall with grasping compensation. Slip state is detected and compensated by closing the gripper
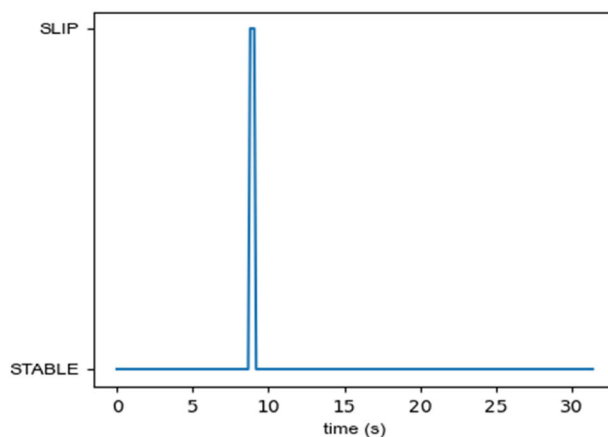
(a) Cardboard

(b) Plastic

(c) Metal

(d) Glass

Fig. 22 Slip detection graphic for each object. These plots show that slip is produced only once after compensating the grasping opening

compensation is applied, only one slip event is detected and the object does not fall. We did not use any compensation algorithm that the software of the gripper has, but instead, we created our own compensation algorithm.

In order to lift the object, our system first needs to know whether or not the object is being grasped. An object is considered to be grasped when a certain number of contact detections are completed. We, therefore, ran the entire perception and tactile system with each one of the four objects, using different numbers of contact detections. The decision as to which number was the most suitable for each object

was made by calculating the same graphics as those shown in Fig. 20. Figure 22 shows that a stable grasp can be achieved for the four objects with a threshold of 3 contact detections for the objects of cardboard, plastic and metal, and 4 contact detections for the glass object because they usually weigh more.

Once the parameters had been established, the final experiment could be carried out. The results are expressed as the accuracy value that denotes the percentage of successful graspings, which we denominate as the Collection Success Rate (CSR).
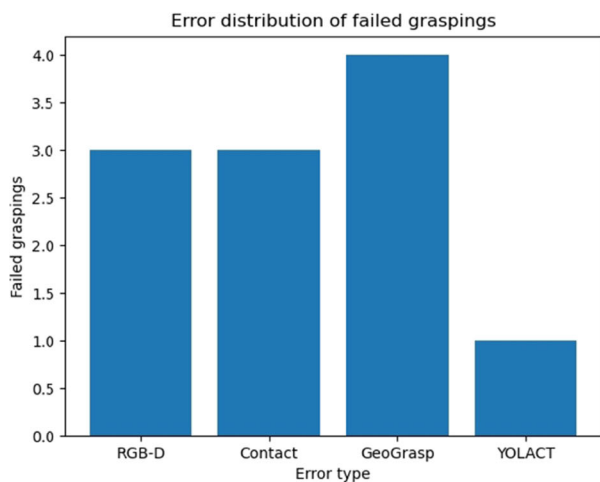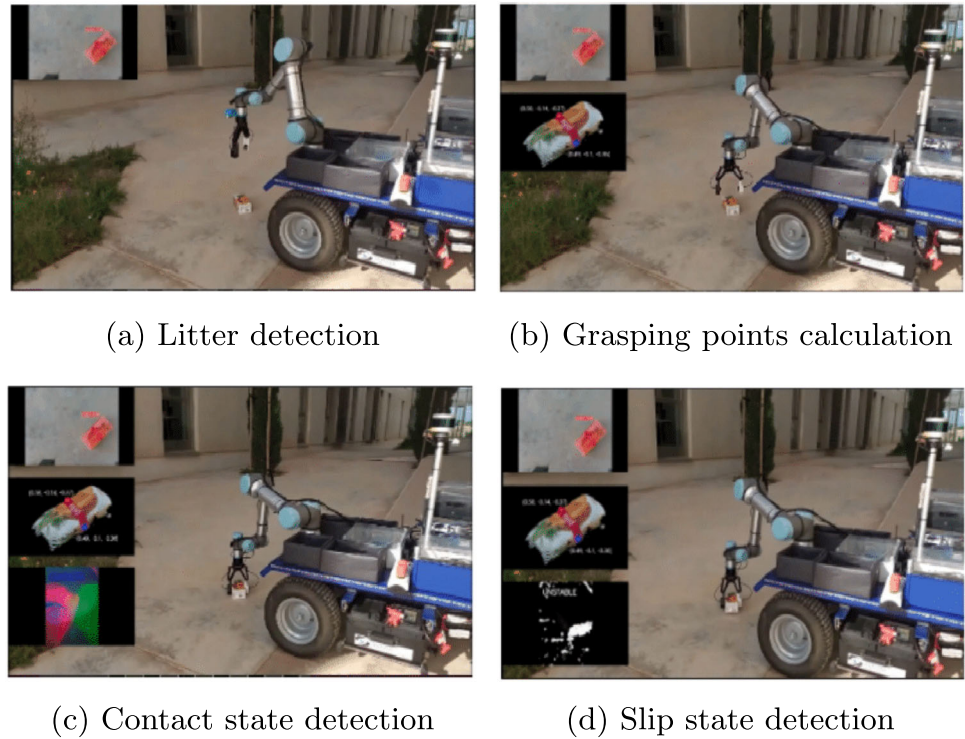
Table 11 CSR values obtained splitting the results in terms of the environment (CSR-Env). These values are between 0 and 1

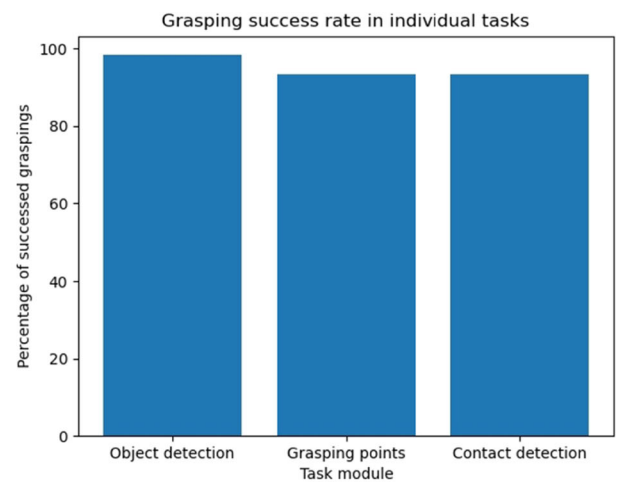|	Tiled pavement	Surface of stone/soil	Grass
CSR-Env	0.80	0.75	0.85

Table 12 CSR values obtained after splitting the results in terms of the object class (CSR-Obj). These values are between 0 and 1

	Cardboard	Plastic	Metal	Glass
CSR-Obj	0.93	0.8	0.87	0.6

Fig. 23 Example of picking up a cardboard object in a tiled pavement environment

(a) Litter detection

(b) Grasping points calculation

(c) Contact state detection

(d) Slip state detection

(a)

(b)

Fig. 24 a) Error type distribution that led to picking up failures. Error types are 1) bad reconstruction from the RGBD camera, 2) failed contact detection, 3) wrong location of the grasping points, and 4) wrong object detection. b) CSR when considering only the errors produced by each individual module

A grasping is considered successful when it achieves all of the following conditions: the object is segmented correctly, the grasping points are located on the object, the contact is detected and the robot is able to lift and move the object to the corresponding box, without dropping it. The success of collecting an object at the first attempt is 80%. However, a failed first attempt can be successful in the second one since grasps are independent probabilistic events. That is, the pipeline is launched again if needed and the object could have changed its position when falling to the floor on the previous attempt.

As can be seen in Table 11, the overall collecting success rate varied from 75% in the worst scenario to 85% in the best one. Our system works better on flat surfaces (tiled pavements) or on surfaces that the gripper can cross in order to grasp the object (grass). Nonetheless, the system also attains promising results on irregular surfaces on which the gripper could collide with the ground. Good results are also obtained when using the same coloured object and environment (metal object and grass environment).

With regard to the results in terms of object class (see Table 12), our system attained a CSR of 60% in the worst case and 93% in the best one, when picking up the objects at the first attempt. The drop in the CSR that occurred with glass objects was owing to their transparency, which makes object segmentation and grasping point calculation more difficult. The grasping pose is not, therefore, precise and the gripper fails when picking up the object.

Figure 23 shows each step of an example of one litter collection attempt with a cardboard object in a tiled pavement environment. In this example our system detects and locates the item of litter as cardboard class, calculates the grasping pose correctly, and performs the manipulation tasks in order to collect the litter.

The most common errors are produced as a result of the wrong locations of the grasping points, failed contact detections, low quality of the point clouds produced by the RGBD camera, or wrong object detection and segmentation. Our system failed at the first attempt on 18% of occasions. Figure 24-a shows the distribution of error types that led to the collection failure. Figure 24-b shows the CSR of each module independently.

5 Discussion and conclusions

In this work, we present a robotic system that is able to collect household objects in outdoor and natural environments. This system was created by implementing three main modules: litter detection, grasping point calculation, and tactile manipulation. These three modules together allow our system to obtain promising and reliable results with unknown objects and in outdoor scenarios.

We have implemented two detection and control methods: one for contact detection, which is based on CNN, in order to know when the object is being grasped, and another for the slip detection, which is based on morphological operations, in order to readjust the gripper opening. Moreover, we have used well-known neural networks for the recognition and segmentation of objects. The result is then combined with 3D techniques to extract the surface features of those objects. Also, we have carried out extensive and rigorous experimentation in order to adjust our visual model to different objects in different scenarios. We first carried out offline experimentation (Sect. 4.2, 4.3 and 4.4) with real recorded data, after which we tested our system in online mode with real-time data (Sect. 4.5). When all the modules work together, our system is able to perform the complex task of picking up household objects in outdoor scenarios by combining the information concerning each module. This is assuming that the biggest object that the system could handle is 118 mm, in which we could manage a rotation error up to 13º with respect to the x_C reference frame system of Fig. 1 - right and a translation error of 3% with respect to the size of the object with respect to the y_C and z_C reference frame systems of Fig. 1 - right (approx 3.5 mm).

Overall, our system performs correctly when collecting unknown objects in new environments, obtaining high accuracy when making more than one attempt. Nonetheless, our system has some limitations that we present as future lines of work. For example, transparent objects cause the detection and grasping point calculation modules several problems resulting from a bad quality point cloud obtained by the RGBD camera. This issue could be solved by a complementary object reconstruction process. Another limitation is the detection of dynamic objects, namely, an object whose position or orientation changes while the detection is taking place, for example, due to the wind. We could solve this by performing a tracking operation in order to update the object pose in real time. We are currently resolving this issue by executing the waste detection in a loop in order to update the object's pose.

Author Contributions I.L.P and S.T.P contributed to the design and implementation of the vision system. J.C.A and P.G contributed to the

design and implementation of the tactile system. I.L.P and J.C.A carried out the experiments in real environments. All authors analysed the results and wrote the manuscript.

Availability of data and materials The dataset D1 designed and used is publicly available at HOWA_dataset. Datasets D2 and D3 designed and used are not publicly available in this moment but they could be available from Institutional Repository of the University of Alicante or asking authors.

Compliance with ethical standards

Conflict of interest The authors declare that they have no known competing finantial interests or personal relationships that could have appeared to influence the work reported in this paper.

Ethics approval   Not applicable.

Consent to participate   Not applicable.

Consent for publication   Authors gives the Publisher the permission to publish the Work in this journal.

Code availability   Not applicable.

References

1. Chiang, C.-H.:Vision-based coverage navigation for robot trash collection task. In: IEEE Int. Conf. on Advanced Robotics and Intelligent Systems (ARIS),pp. 1–6 (2015). https://doi.org/10.1109/ARIS.2015.7158229.IEEE

2. Muthugala, M.V.J., Samarakoon, S.B.P., Elara, M.R.: Tradeoff between area coverage and energy usage of a self-reconfigurable floor cleaning robot based on user preference. IEEE Access **8**, 76267–76275 (2020). https://doi.org/10.1109/ACCESS.2020.2988977

3. Zapata-Impata, B.S., Shah, V.,Singh, H.,Platt, R. :Autotrans: an autonomous open world transportation system. arXiv:1810.03400 (2018).https://doi.org/10.48550/arXiv.1810.03400

4. Sun, C., Orbik, J., Devin, C.M., Yang, B.H., Gupta, A., Berseth, G., Levine, S. :Fully autonomous real-world reinforcement learning with applications to mobile manipulation. In: 5th Conf. on Robot Learning (CoRL) (2021). https://doi.org/10.48550/2107.13545

5. Sultana, R., Adams, R.D., Yan, Y., Yanik, P.M., Tanaka, M.L. :Trash and recycled material identification using convolutional neural networks (cnn).In: SoutheastCon,pp. 1–8 (2020).https://doi.org/10.1109/SoutheastCon44009.2020.9249739

6. Lin, Y., Sun, Y.: Robot grasp planning based on demonstrated grasp strategies. Int. J. Robot. Res. **34**(1), 26–42 (2015). https://doi.org/10.1177/0278364914555544

7. Zapata-Impata, B., Gil, P., Pomares, J., Medina, F. :Fast geometry-based computation of grasping points on three-dimensional point clouds.International Journal of Advanced Robotic Systems 16(2019). https://doi.org/10.1177/1729881419831846

8. del Pino, I., Muñoz-Bañon, M.Á., Cova-Rocamora, S., Contreras, M.Á., Candelas, F.A., Torres, F.: Deeper in blue. Journal of Intelligent & Robotic Systems **98**, 207–225 (2020). https://doi.org/10.1007/s10846-019-00983-6

9. Lambeta, M., Chou, P.-W., Tian, S., Yang, B., Maloon, B., Most, V.R., Stroud, D., Santos, R., Byagowi, A., Kammerer, G., Jayaraman, D., Calandra, R.: Digit: A novel design for a low-cost compact high-resolution tactile sensor with application to in-hand manipulation.IEEE Robotics and Automation Letters **5**(3),3838–3845 (2020). https://doi.org/10.1109/LRA.2020.2977257

10. Chandra, S.S., Kulshreshtha, M., Randhawa, P. :A review of trash collecting and cleaning robots. In: 9th Int. Conf. on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), pp. 1–5 (2021). https://doi.org/10.1109/ICRITO51393.2021.9596551

11. Bai, J., Lian, S., Liu, Z., Wang, K., Liu, D.: Deep learning based robot for automatically picking up garbage on the grass. IEEE Transactions on Consumer Electronics **64**(3), 382–389 (2018). https://doi.org/10.1109/TCE.2018.2859629

12. Liu, J., Balatti, P., Ellis,K., Hadjivelichkov, D., Stoyanov,D., Ajoudani, A., Kanoulas, D. :Garbage collection and sorting with a mobile manipulator using deep learning and whole-body control. In: IEEE 20th Int. Conf. on Humanoid Robots (Humanoids),pp. 408–414 (2021).https://doi.org/10.1109/HUMANOIDS47582.2021.9555800

13. Mnyussiwalla, H., Seguin, P., Vulliez, P., Gazeau, J. :Evaluation and selection of grasp quality criteria for dexterous manipulation.Journal of Intelligent & Robotic Systems104, 20 (2022).https://doi.org/10.1007/s10846-021-01554-4

14. ten Pas, A., Gualtieri, M., Saenko, K., Platt, R. :Grasp pose detection in point clouds.The International Journal of Robotics Research **36**(13-14),1455–1473 (2017). https://doi.org/10.1177/2F0278364917735594

15. Dong, S., Yuan, W., Adelson, E.H. :Improved gelsight tactile sensor for measuring geometry and slip.In: IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS),pp. 137–144 (2017). https://doi.org/10.1109/IROS.2017.8202149

16. Yuan, W., Dong, S., Adelson, E.H.: Gelsight: High-resolution robot tactile sensors for estimating geometry and force. Sensors **17**(12), 2762 (2017). https://doi.org/10.3390/s17122762

17. Zhang, Y., Yuan, W., Kan, Z., Wang, M.Y. :Towards learning to detect and predict contact events on vision-based tactile sensors. In: 3rd Conf. on Robot Learning (CoRL), pp. 1395–1404 (2019). https://doi.org/10.48550/arxiv.1910.03973

18. Zhang, Y., Kan, Z., Tse, Y.A., Yang, Y., Wang, M.Y. :Fingervision tactile sensor design and slip detection using convolutional lstm network. (2018).https://doi.org/10.48550/arXiv.1810.02653

19. James, J.W., Pestell, N., Lepora, N.F. :Slip detection with a biomimetic tactile sensor.IEEE Robotics and Automation Letters **3**(4),3340–3346 (2018).https://doi.org/10.1109/LRA.2018.2852797

20. Li, J., Dong, S., Adelson, E. :Slip detection with combined tactile and visual information. In: IEEE Int. Conf. on Robotics

and Automation (ICRA), pp. 7772–7777 (2018).https://doi.org/10.1109/ICRA.2018.8460495

21. James, J.W., Lepora, N.F.: Slip detection for grasp stabilization with a multifingered tactile robot hand. IEEE Transactions on Robotics **37**(2), 506–519 (2021). https://doi.org/10.1109/TRO.2020.3031245

22. Tornero, P., Puente, S., Gil, P. :Detection and location of domestic waste for planning its collection using an autonomous robot. In: IEEE 8th Int. Conf. on Control, Automation and Robotics (ICCAR), Xiamen, China, pp. 138–144 (2022). https://doi.org/10.1109/ICCAR55106.2022.9782609

23. Hafiz, A.M., Bhat, G.M.: A survey on instance segmentation: state of the art. International Journal of Multimedia Information Retrieval **9**(3), 171–189 (2020). https://doi.org/10.1007/s13735-020-00195-x

24. Gu, W., Bai, S., Kong, L.: A review on 2d instance segmentation based on deep neural networks. Image and Vision Computing **120**,(2022). https://doi.org/10.1016/j.imavis.2022.104401

25. He, K., Gkioxari, G., Dollar, P., Girshick, R. :Mask r-cnn.In: IEEE/CVF Int. Conf. on Computer Vision (ICCV) (2017). https://doi.org/10.48550/arXiv.1703.06870

26. Bolya, D., Zhou, C., Xiao, F.,Lee, Y.J. :Yolact: Real-time instance segmentation. In: IEEE/CVF Int. Conf. on Computer Vision (ICCV), pp. 9157–9166 (2019). https://doi.org/10.48550/arXiv.1904.02689

27. Bolya, D., Zhou, C., Xiao, F., Lee, Y.J. :Yolact++ better real-time instance segmentation.IEEE Transactions on Pattern Analysis and Machine Intelligence **44**(2),1108–1121 (2022). https://doi.org/10.1109/TPAMI.2020.3014297

28. Ren, S., He, K., Girshick, R., Sun, J. :Faster r-cnn: Towards real-time object detection with region proposal networks.IEEE Transactions on Pattern Analysis and Machine Intelligence **28**(6),1137–1149 (2015).https://doi.org/10.1109/TPAMI.2016.2577031

29. He, K., Zhang, X., Ren, S., Sun, J. :Deep residual learning for image recognition. In: IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) (2016).https://doi.org/10.1109/CVPR.2016.90

30. Redmon, J. :Darknet: Open Source Neural Networks in C.http://pjreddie.com/darknet/ (2013–2016)

31. De Gea, V., Puente, S.T., Gil, P. :Domestic waste detection and grasping points for robotic picking up. In: IEEE Int. Conf. on Robotics and Automation (ICRA)-Workshop: Emerging Paradigms for Robotic Manipulation: from the Lab to the Productive World (2021). https://doi.org/10.48550/arXiv.2105.06825

32. Garrido-Jurado, S., Muñoz-Salinas, R., Madrid-Cuevas, F.J., Mar'ın-Jiménez, M.J. :Automatic generation and detection of highly reliable fiducial markers under occlusion.Pattern Recognition **47**(6),2280–2292 (2014).https://doi.org/10.1016/j.patcog.2014.01.005

33. Coleman, D., Sucan, I., Chitta, S., Correll, N. :Reducing the barrier to entry of complex robotic software: a moveit! case study. Journal of Software Engineering for Robotics **5**(1), 3–16 (2014). https://doi.org/10.48550/arXiv.1404.3785

34. LaValle, S.M. :Rapidly-exploring random trees: A new tool for path planning.Technical Report 11,Computer Science Dept., Iowa State University (October 1998). http://lavalle.pl/papers/Lav98c.pdf

35. LaValle, S.M., Kuffner Jr, J.J. :Randomized kinodynamic planning.The International Journal of Robotics Research **20**(5), 378–400 (2001). https://doi.org/10.1177/02783640122067453

36. Ward-Cherrier, B., Pestell, N., Cramphorn, L., Winstone, B., Giannaccini, M.E., Rossiter, J., Lepora, N.F.: The tactip family: Soft optical tactile sensors with 3d-printed biomimetic morphologies. Soft robotics **5**(2), 216–227 (2018). https://doi.org/10.1089/soro.2017.0052

37. Pagoli, A., Chapelle, F., Corrales-Ramon, J.-A., Mezouar, Y., Lapusta, Y.: Large-area and low-cost force/tactile capacitive sensor for soft robotic applications. Sensors **22**(11), 4083 (2022). https://doi.org/10.3390/s22114083

38. Kappassov, Z., Corrales, J.-A., Perdereau, V.: Touch driven controller and tactile features for physical interactions. Robotics and Autonomous Systems **123**, 103332 (2020). https://doi.org/10.1016/j.robot.2019.103332

39. Castaño Amorós, J., Gil, P., Puente Méndez, S.T. :Touch detection with low-cost visual-based sensor. In: 2nd Int. Conf. on Robotics, Computer Vision and Intelligent Systems (ROBOVIS), pp. 136–142 (2021).https://doi.org/10.5220/0010699800003061

40. Simonyan, K., Zisserman, A. :Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv (2014). https://doi.org/10.48550/arXiv.1409.1556

41. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z. :Rethinking the inception architecture.In: Conf. on Computer Vision and Pattern Recognition (CVPR) (2016). https://doi.org/10.1109/CVPR.2016.308

42. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.-C. :Mobilenetv2: Inverted residuals and linear bottlenecks.In: IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) (2018).https://doi.org/10.48550/arXiv.1801.04381

43. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. International Journal of Computer Vision **88**(2), 303–338 (2010). https://doi.org/10.1007/s11263-009-0275-4

44. Salton, G., McGill, M.J. :Introduction to Modern Information Retrieval.McGraw-Hill, Inc.,USA (1986). https://dl.acm.org/doi/book/10.5555/576628

45. Hossin, M., Sulaiman, M.N.: A review on evaluation metrics for data classification evaluations. International Journal of Data Mining & knowledge management process **5**(2), 1 (2015). https://doi.org/10.5121/ijdkp.2015.5201

46. Torralba, A., Russell, B.C., Yuen, J.: Labelme: Online image annotation and applications. Proceedings of the IEEE **98**(8), 1467–1484 (2010). https://doi.org/10.1109/JPROC.2010.2050290

47. Kingma, D.P., Ba, J. :Adam: A method for stochastic optimization. In: 3rd Int. Conf. for Learning Representations (ICLR) (2014). https://doi.org/10.48550/arXiv.1412.6980

Ignacio de Loyola Páez-Ubieta obtained his Bachelor's and Master's Science degrees in Robotics Engineering and Automation and Robotics Engineering from the University of Alicante (Spain) in 2020 and 2021, respectively. He worked as a research technician at the Automation, Robotics and Computer Vision Group (AUROVA) of the same university. He is currently studying for his Ph. D. in Computer Science at the aforementioned university within the same research group, while also working as a predoctoral researcher for Miguel Hernandez University of Elche (Spain). He has been awarded numerous prizes during his academic career, such as the National Ranking of Academic Excellence 2020 (due to his B. S. degree) and the Extraordinary Award for the Best Academic Record 2020/2021 (due to his M. S. degree). His research interests comprise robotics and deep learning, in particular demonstration-based and imitation learning.

Julio Castaño-Amorós received the Bachelor and Master degrees in Robotics from the University of Alicante, Spain, in 2020 and 2021, respectively. He is currently studying a Ph.D. degree as a researcher in the Automation, Robotics and Computer Vision Group (AUROVA) supported by the University of Alicante. His research is mainly focused on the intersection of deep learning techniques and robotic manipulation based on tactile sensors.

Santiago T. Puente received the Computer Science Engineer and the PhD degrees in the University of Alicante (Spain), in 1998 and 2003 respectively. He is a full-time lecture and researcher at the University of Alicante since 2003, where he teaches currently in the Degree in Robotic Engineering. From 2013 to 2019, Dr. Puente has been deputy director of Infrastructures and facilities Polytechnic School of the University of Alicante, Furthermore, from 2019 to 2021 he has been Academic Coordinator of BEng Robotics Engineering. Dr. Puente also researches in the Automation, Robotics and Computer Vision Group (AUROVA) of the University of Alicante since 1999, and he has involved in several research projects and networks supported by the Spanish Government, as well as development projects in collaboration with regional industry. His research interests include automation and robotics (intelligent robotic manipulation, robot perception systems, robot imitation learning, field mobile robots), and e-learning. Currently, his research focuses on robotics, and e-learning.

Pablo Gil received a B.Sc. degree in Computer Science Engineering and a Ph.D. degree from the University of Alicante, Spain, in 1999 and 2008, respectively. He was assistant professor from 2009 to 2021 in the Department of Physics, System Engineering and Signal Theory at University of Alicante. Currently, he holds a tenured position as Full Professor of Automatic Control. From 2019 to 2023, he was the head of Computer Science Research Institute in Alicante. He has also been a Researcher on over 20 research and development projects funded by the European Commission, Spanish Government agencies and private companies. His research interests include computer vision, 3-D vision, robotic manipulation and tactile perception. He has co-authored more than 100 works on those topics. Dr. Gil is a member of the Spanish Automatic Committee of IFAC and a senior member of the IEEE Robotics and Automation Society, being secretary of the Spanish Chapter from 2018 to 2022.