SN Computer Science (2023) 4:429
https://doi.org/10.1007/s42979-023-01927-6

ORIGINAL RESEARCH

Grasp Pose Estimation for Pick and Place of Frozen Blood Bags Based on Point Cloud Processing and Deep Learning Strategies Using Vacuum Grippers

Muhammad Zain Bashir[1] · Jaeseok Kim[1] · Olivia Nocentini[1,2] · Filippo Cavallo[1,2]

Received: 23 December 2022 / Accepted: 18 May 2023

Abstract
We describe different strategies to compute grasp poses for vacuum grippers for pick and place of frozen blood bags. Our methods process RGB-D data to search for local flat patches on the bags' surface which act as grasp points when using vacuum grippers. We develop three strategies which analyze point cloud data to propose gripper poses and one method that trains a real-time object detector to propose the grasp point and processes point cloud data to compute the bag's orientation. All the strategies are based on the computation of a normal vector at each 3D point to account for the surface orientation. They differ from each other based on how each method searches for these flat patches. We validate and compare the effectiveness of our methods by conducting real-world pick and place experiments, achieving an average success rate of above 80%. In conclusion, four different strategies, both analytical and a hybrid of analytic and deep learning approaches to infer optimal grasp poses for vacuum grippers to automatise pick and place operations of blood bags were presented.

Keywords Vacuum grippers · Grasp pose · Object detector · Point cloud · Deep learning

Introduction

A very time-consuming task in healthcare industry (hospitals, analytical laboratories, and blood banks) is the handling of frozen blood bags. These tasks include sorting according to blood groups, blood components or patients, and donors or transportation from one freezer to another, etc. These simple yet repetitive tasks consume a lot of workers' time. A valid solution to circumvent this problem is integrating an automatic pick and place system to manage these operations. While there exist many robotic solutions for pick and place tasks in industrial settings, they are almost always problem-specific which makes it challenging to adopt them to other scenarios.

A promising approach to automating the handling of blood bags is using vision-based blood bag detection with a commercial robotic arm fitted with a vacuum gripper. Over the past decade, with the rapid advancement in vision-based sensor technology, computer vision has become part and parcel of the widely investigated field of robotics in general and robotic manipulation in particular. Vision-based detection methods allow for an object's position detection in pixel coordinates while RGB-D cameras help to transform these coordinates to 3D world coordinates. In spite of this interest in deploying vision to solve robotic manipulation problems, tons of challenges still remain that stem from the object's properties, such as color and deformation, cluttered environments or occluded/partial views of the scene. The challenges thus posed by handling of frozen blood bags arise from them developing irregular and curved surfaces (Fig. 1) on freezing. Added to this problem is the bags' surfaces going watery and slippery as soon as they are removed from the freezer. Simultaneously, using multi-fingered end-effectors

✉ Olivia Nocentini
olivia.nocentini@unifi.it; olivia.nocentini@santannapisa.it

Muhammad Zain Bashir
muhammadzain.bashir@unifi.it

Jaeseok Kim
jaeseok.kim@unifi.it

Filippo Cavallo
filippo.cavallo@unifi.it

[1] Department of Industrial Engineering, University of Florence, Via di Santa Marta 3, Florence 50139, Tuscany, Italy

[2] Biorobotics Institute, Scuola Superiore Sant'Anna Pisa, Viale Rinaldo Piaggio 34, Pontedera (PI) 56025, Tuscany, Italy

Published online: 07 June 2023

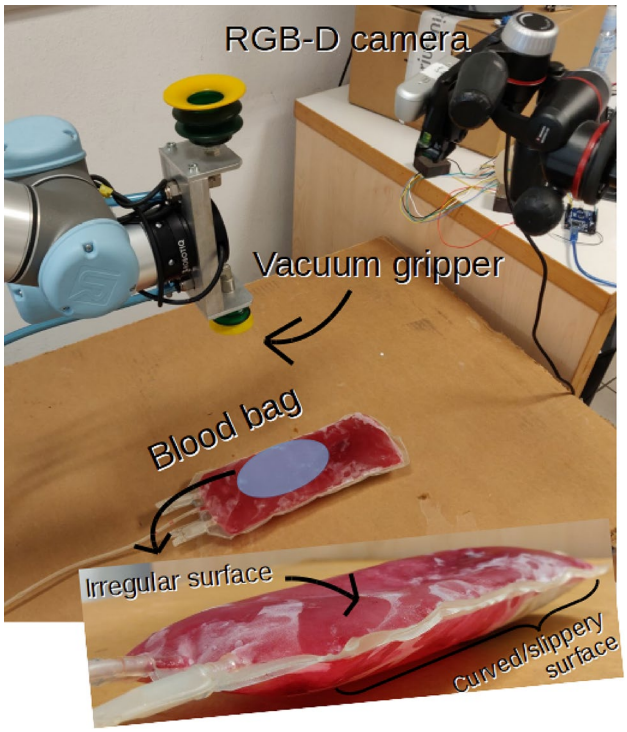

Fig. 1 Experiment setup: The vacuum gripper, RGB-D camera, and the blood bag. The bag develops curved and irregular surfaces on freezing

to pick and place the bags results in complex grasp pose computations. These calculations need sufficient amount of computing power to compute grasps efficiently [1]. Sometimes environmental constraints, such as cluttered and narrow spaces, make use of multi-fingered end-effectors difficult while using vacuum grippers sidesteps these difficulties by simplifying the grasp motion [2]. Using vacuum grippers results in a much less complex grasp computation as only one position and one orientation calculation are required. It is easy to adapt such grasp systems to various tasks, making suction grasping one of the widely used grasping strategies in industrial settings.

In view of this, we develop automating the pick and place operations of frozen blood bags for healthcare industry. We use a vision system and a vacuum gripper attached to a commercial robotic manipulator to complete the desired task. Data from an RGB-D camera are used to compute both the blood bag's orientation and position while the vacuum gripper is used to handle the bags. We develop four different strategies for the grasp pose computation. The first three, namely the ***Normal error***, ***Z-error***, and the ***Average surface curvature change*** are model-based traditional vision approaches to computing the grasp pose while the last one, the ***YOLO grasp***, is a data-driven approach based on the real-time object detector YOLO [3]. Concretely, our contributions are the following:

- We propose four strategies that use vision to compute grasp pose for vacuum grippers.
- We perform real-world experiments in order to test, validate, and compare these strategies' effectiveness in a real-world scenario.
- We construct a blood bag dataset for training comprising approximately of 15000 images with their bounding box annotations. The bags vary in size, color, and the number of bags per image and are present under different background and lighting conditions. Data augmentation is applied to upscale the dataset.

Related Work

Over the past years, there have been many works that address the issue of computing optimal grasp poses for vacuum grippers using vision-based solutions. The basic idea behind these works is to search for flat patches on the object surfaces since these flat patches inherently present themselves as good grasp candidates when using vacuum grippers. These vision-based methods can be broadly classified as either being machine/deep learning-based or model-based. In most cases, the latter refers to using 3D vision and point cloud processing to infer optimal grasp poses.

Deep Learning for Grasp Point Computation for Vacuum Grippers

In using deep learning to find optimal grasp poses, [4] propose a Deep Convolutional Neural Network (DCNN) that can predict grasp points on texture-less planar surfaces, to be picked by vacuum grippers. Their network can also predict the activation pattern for the two suction cups that they use. The authors also extract surface features (normal and center position) to refine the predicted grasp point position. However, their approach only focuses on planar and texture-less surfaces while requiring huge amount training data, being a supervised learning-based approach. The authors of [5] construct a dataset called **Dex-net 3.0** to train a Grasp Quality Convolutional Neural Network (GQCNN) to classify the robustness of suction targets based on a compliant suction model. This model rates this robustness on the ability of the proposed targets to resist external forces. Building on top of this work [6] trained a **YOLO** [3] object detector to identify envelopes in an automatic package dispatching facility. The authors choose the optimal grasp point as a 3D coordinate near the center of the predicted bounding box according to the suction sampling policy described in [5]. Another work, [7], focus on refining depth values for transparent object surfaces using what they term the **ClearGrasp** algorithm. They first use a CNN developed in [8] to predict surface normals, segmentation masks and occlusion boundaries

and then refine the noisy depth estimates using a global optimization algorithm. They validated their algorithm in a real-world robotic picking system by using a suction and a parallel jaw gripper but their approach focuses on dealing with transparent objects. In attempting to find grasp poses for suction grippers [9] successfully deployed the **U-net** CNN developed in [10] to output a suction region probability map. Each pixel value denotes the probability of a successful grasp if a suction gripper pushes down vertically on the 3D point corresponding to a given 2D pixel. They tested their method to complete a bin-picking task. However, their approach doesn't take the objects' orientation into account. [11] computed robust suction affordances based on two CNNs: a Fast Region Estimation Network (**FRE-Net**) that identifies regions containing pickable objects while a Suction Grasp point affordance Network (**SGPA-Net**) predicts the best pickable point in that region. Shukla et al [12] took a different approach to using deep learning to find grasp poses. They developed the Grasp Deep Q-Learning (GDQN) algorithm in which they used a Deep reinforcement learning paradigm to learn grasp orientation for robotic grippers. They modeled their state as an RGB image cropped using bounding box coordinates obtained using the YOLO [3] object detector. A sparse reward function was used while the gripper was allowed to take discreet actions in the form of angular rotations. Their experiments showed the proposed architecture allowed better orientation learning performance over the MVGG16 architecture, a modified version of the VGG16 network.

Point Cloud Processing for Grasp Point Computation for Vacuum Grippers

Model-based approaches to compute grasp poses for suction grippers have long been in use. [13] and [14] use suction grippers to handle meat pieces in a meat processing plant. They use a region growing algorithm to segment out the meat piece from the point cloud data and apply Principal Component Analysis (PCA) to 3D points and project them onto the 2D PCA frame. This is followed using a concave hull to determine the edges of the meat piece and the suction cups are placed within the meat boundary based on a distance measure between the cups and the meat boundary. In [15], the authors design a three-fingered hand that includes a suction cup as the third finger. They use a Fully Convolutional Network called FCN [16] to segment objects in RGB images. They then extract the corresponding point cloud, performing euclidean clustering and computing its centroid as the grasp point. One important model-based strategy to identify grasp point for vacuum grippers is presented in [1]. They first design a four suction cup configurable gripper with an RGB-D camera attached to the robot's end-effector. Afterwards, they compute the grasp pose for their gripper by performing a normal sphere analysis to cluster the surface normals oriented in similar directions. The points resulting from this analysis are projected onto a 2D plane and the algorithm analyses these projections. The outcome is the optimal configuration of the gripper. A final correction of the grasp pose is then calculated by computing a rigid transform, using Singular Value Decomposition, between the reference points of all suction cups and the estimated point of contact. Kim et al. [17] have used point cloud segmentation to extract planes and have chosen the best grasp point as the centroid of the segmented plane in a bin picking and sorting task.

Methodology

Each of the following "Preprocessing", "Normals Computation", "Orientation Computation", and "Normal Error (NE)" provide details on the grasp point computation for each strategy employed and the common processing pipeline is presented here which forms the backbone of the orientation computation, as shown in Fig. 2. While *Normal error*, *Z-error* and *Average surface curvature change* only require point cloud processing to find local flat patches on the bag's surface, *YOLO grasp* also requires an RGB image to compute the grasp pose. For point cloud processing, we assume that the blood bag lays on a flat surface. After computing the grasp pose's position component, it's orientation is computed followed by the computation of the transformation between the camera coordinates and the coordinate frame assigned to the bag.

Preprocessing

The RANSAC algorithm [18] is first used to compute the coefficients of a plane model corresponding to the table top used for experiments. Then, all the 3D points that satisfy the plane model are removed. Downsampling is then applied using a voxel grid filter. A cube size of 5 mm is chosen and all the 3D points present in a given grid are approximated with their centroid. Thus, the number of points is reduced. To further simplify computations, 75% of the 3D points furthest away from the camera are discarded. This removes points which are below the bag's edges and restricts the grasp point search to the bag's top surface. Euclidean clustering is performed on the remaining points to segment out the bag and the largest cluster is selected ensuring that only the points corresponding the bag are kept. A point's membership to a cluster depends on the chosen distance threshold, d_{th}. A cluster $O_i = \{\mathbf{p}_i \in P\}$ is distinct from another cluster $O_j = \{\mathbf{p}_j \in P\}$ if:

$$\|\mathbf{p}_j - \mathbf{p}_i\|_2 > d_{th} \tag{1}$$

where $\|\mathbf{p}_j - \mathbf{p}_i\|_2$ is the euclidean distance between the points $\mathbf{p}_i$ and $\mathbf{p}_j$.

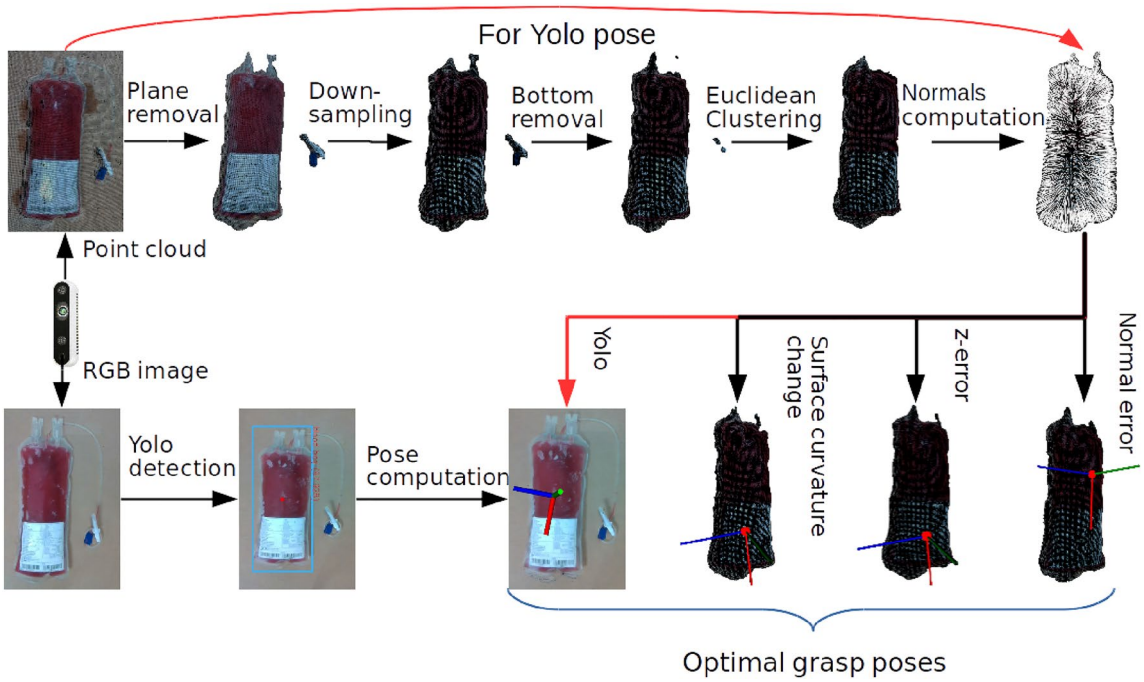

Fig. 2 Block diagram of the processing pipeline that results in grasp pose computation for all strategies

The bag cluster is selected by computing the neighbors of each point $\mathbf{p}_i$ within a sphere of radius $r_c < d_{th}$. A spatial relationship is simultaneously maintained such that two points belonging to the same cluster can be traced through any of their neighbors.

Normals Computation

All four strategies require the computation of the normal vector for each point in the point cloud in order to establish the pose of the bag. The problem of determining the normal to a point on the surface is approximated by the problem of estimating the normal of a plane tangent to the surface, which in turn becomes a least-square plane fitting estimation problem in $\mathbf{P}^k$, the set of neighbors of a query point $\mathbf{p}_q$. The normal vector $\mathbf{n} = \{n_x, n_y, n_z\}$ at $\mathbf{p}_q$ is computed by analyzing the eigenvalues and eigenvectors of the covariance matrix $C \in \mathbb{R}^{3x3}$ of $\mathbf{P}^k$, expressed as:

$$C = \frac{1}{k}\sum_{i=1}^{k}(\mathbf{p}_i - \bar{\mathbf{p}}) \cdot (\mathbf{p}_i - \bar{\mathbf{p}})^{\mathsf{T}}, C \cdot \mathbf{v}_j = \lambda_j \cdot \mathbf{v}_j \tag{2}$$

where k is the index of a neighborhood point in the set $\mathbf{P}^k$, $j \in \{0, 1, 2\}$, $\mathbf{v}_j$ and λ_j are eigenvectors and eigenvalues of C respectively and $\bar{\mathbf{p}}$ is the centroid of $\mathbf{P}^k$ computed as:

$$\bar{\mathbf{p}} = \frac{1}{k} \cdot \sum_{i=1}^{k} \mathbf{p}_i \tag{3}$$

Orientation Computation

A pose calculation can be broken down into two parts: orientation computation and position computation. The bag's orientation is calculated using the normal vectors calculated earlier, whereas the position of the grasp point is derived from one of the four strategies explained in the following sections. Once the position is computed, a coordinate frame is assigned to the bag at that position with its y-axis aligned with the normal vector at that point. The choice of y-axis alignment is arbitrary. A 3D rigid transformation between the camera frame and the bag frame is then calculated. The translation part of the transformation is straightforward since it corresponds to the position of the bag in the camera frame. An angle–axis representation is used to represent the rotation between the two frames. The rotation angle θ_r is calculated as:

$$\theta_r = \arccos(\mathbf{n}_b \cdot \hat{\mathbf{y}}_c) \tag{4}$$

where $\mathbf{n}_b$ is the normal vector at the computed grasp position on the bag's surface and $\hat{\mathbf{y}_c}$ is the unit y-axis of the camera coordinate system. Simultaneously, the normalized rotation axis, $\hat{\mathbf{a}}_r$, is computed as:

$$\hat{\mathbf{a}}_r = \frac{\mathbf{n}_b \times \hat{\mathbf{y}}_c}{\|\mathbf{n}_b \times \hat{\mathbf{y}}_c\|} \tag{5}$$

where $\times$ signifies a cross product and $\|\mathbf{n}_b \times \hat{\mathbf{y}}_c\|$ is the norm of the cross product. Finally, the angle-axis representation is converted into a quaternion $\mathbf{q}$:

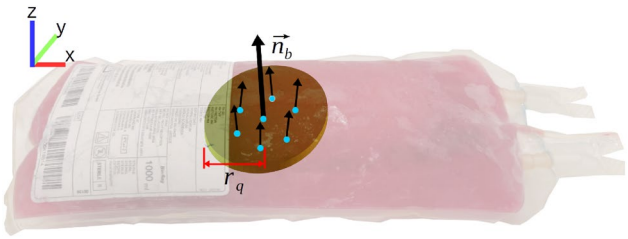

Fig. 3 Query patch, neighboring points and the reference vector to compute the Normal error

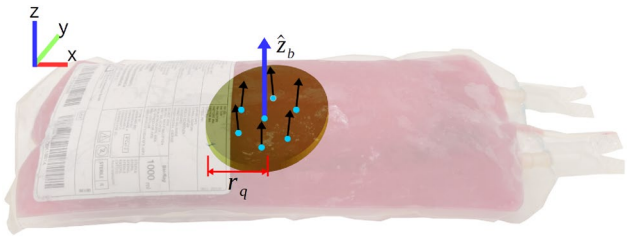

Fig. 4 Query patch, neighboring points and the reference vector to compute the Z-error

$$\mathbf{q} = \begin{bmatrix} \mathbf{q}_w \\ \mathbf{q}_x \\ \mathbf{q}_y \\ \mathbf{q}_z \end{bmatrix} = \begin{bmatrix} \cos(\theta_r/2) \\ \hat{\mathbf{a}}_{r_x} \sin(\theta_r/2) \\ \hat{\mathbf{a}}_{r_y} \sin(\theta_r/2) \\ \hat{\mathbf{a}}_{r_z} \sin(\theta_r/2), \end{bmatrix} \tag{6}$$

where $\hat{\mathbf{a}}_{r_x}$, $\hat{\mathbf{a}}_{r_y}$ and $\hat{\mathbf{a}}_{r_z}$ are the x, y and z components of the normalized rotation axis $\mathbf{a}_r$.

Since ROS transformation tree is maintained and the transformation between the robot base and the camera is already known, the quaternion is transformed to the robot base frame for the actual pick and place operation.

Normal Error (NE)

To find local flat patches on the bag's surface *NE* slides a sphere of radius $\mathbf{r}_q$ *cm* through each point in the point cloud and computes an error value for each point, using all the points lying in the sphere, denoted as $\mathbf{P}^k$. This error value at each point, denoted as $\mathbf{e}_n$, is computed as:

$$\mathbf{e}_{n_q} = \frac{1}{k} \sum_{i=1}^{k} \arccos(\mathbf{n}_b \cdot \mathbf{n}_i), \tag{7}$$

where q is the index of the query point in the point cloud, $\mathbf{n}_b$ is the normal vector at the query point as shown in Fig. 3 and $\mathbf{n}_i$ the normal vector at the i^{th} neighborhood point in the sphere of radius $\mathbf{r}_q$ *cm* and k is the total number of points in the neighborhood. The grasp point $\mathbf{O}_g = \mathbf{p}_{\mathbf{q}_{optimal}}$ is then selected based on our optimality criteria on $\mathbf{q}$:

$$\mathbf{q}_{optimal} = \arg\min_{\mathbf{q}}(\mathbf{e}_{n_q}) \,||\, k > \mathbf{k}_{\min}, \tag{8}$$

where $\mathbf{q}_{optimal}$ is the index of the grasp point in the point cloud, $||$ stands for the AND operation and $\mathbf{k}_{min}$ is the minimum number of neighbors required to be considered the best grasp point.

Z-Error (ZE)

The notable difference between *NE* and *ZE* is the selection of the reference to compute the error. *NE* takes as reference the normal vector at the center of the query (Fig. 3) patch while the latter maintains the reference as the unit-z axis, $\hat{\mathbf{z}}_b$, of the local bag frame as shown in Fig. 4. This error value, denoted as $\mathbf{e}_z$, is computed as:

$$\mathbf{e}_{z_q} = \frac{1}{k} \sum_{i=1}^{k} \arccos(\hat{\mathbf{z}}_b \cdot \mathbf{n}_i), \tag{9}$$

where q is the index of the query point in the point cloud.

Thus, this approach aims to find out flat patches on the bag's surface which are parallel the table top. The grasp point selection criteria from (8) applies here as well with $\mathbf{e}_{n_q}$ being replaced by $\mathbf{e}_{z_q}$.

Average Surface Curvature Change (ASCC)

The goal of *ASCC* is to quantify the deviation of a patch on the bag's surface from being a plane. A surface curvature change value of zero would thus mean that the underlying patch is completely flat. The computation of surface curvature change is closely related to the computation of normals mentioned above. A covariance matrix is first constructed based on equation (2). This is followed by an eigen value analysis of the matrix which results in three different eigen values. The surface curvature change, $\boldsymbol{\sigma}_{p_q}$ at the query point $\mathbf{p}_q$ using k neighboring points is given as:

$$\boldsymbol{\sigma}_{p_q} = \frac{\lambda_0}{\lambda_0 + \lambda_1 + \lambda_2}, \tag{10}$$

where λ_0 is the smallest eigenvalue.

As shown in Fig. 5, a curvature change value $\boldsymbol{\sigma}_{p_q}$ for each point in the query neighborhood is calculated and the geometry of the underlying patch is characterized by an *Average surface curvature change* value, denoted as $\boldsymbol{\sigma}_{av}$:

Fig. 5 Query patch, neighboring points and the reference vector to compute ASCC

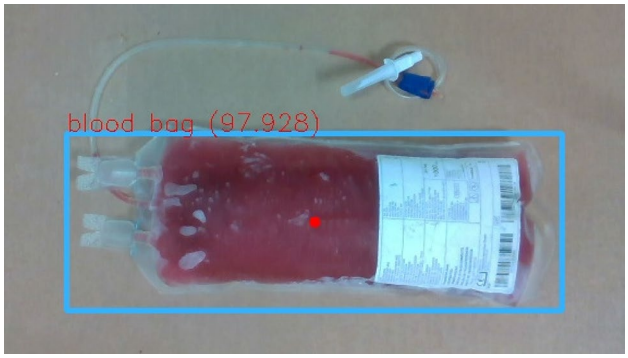

Fig. 6 Bounding box prediction by YOLO. Center of the bounding box is shown by a red colored dot

$$\boldsymbol{\sigma}_{av_q} = \frac{1}{k}\sum_{q=1}^{k} \boldsymbol{\sigma}_{p_q}. \tag{11}$$

The selection of the grasp point follows from the criteria given in Eq. (8) with $\mathbf{e}_{n_q}$ being replaced by $\boldsymbol{\sigma}_{av_q}$. This technique can also be seen as analogous to fitting a small plane in the vicinity of the query point $\mathbf{p}_q$ using a small number of k neighboring points (10 points used in our case) as shown in Fig. 5. The *Average surface curvature change* is then given by the average value of the curvature change of all the small planes in the query patch.

YOLO Grasp (YG)

A YOLO network is trained and used to detect the blood bag in the camera frame. The grasp point then corresponds to the 3D location of pixel coordinates of the center of the bounding box predicted by YOLO as shown by the red dot in Fig. 6. The above described orientation computation pipeline is then followed in order to compute the bag's pose.

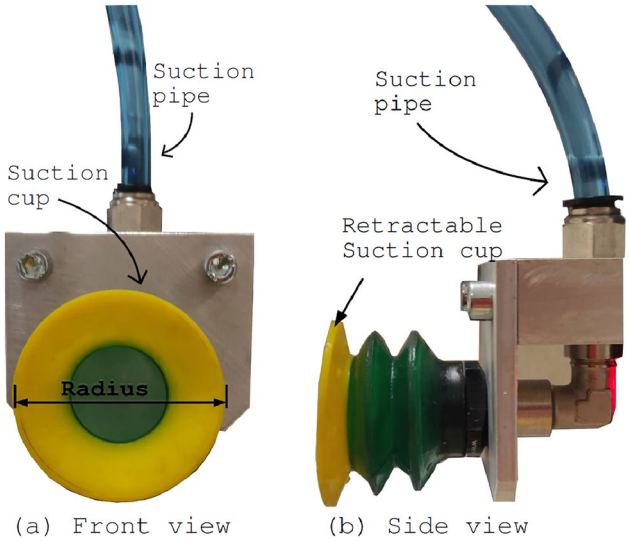

Fig. 7 Vacuum gripper

Experiment Set-up and Details

To realize the pick and place operation of the blood bags, a UR5 Universal Robot arm, equipped with a suction gripper, was set up with a D435 RGB-D camera overlooking the workspace as shown in Fig. 1.

Robot Control

As stated earlier, the Robot operating system (ROS) middleware was used to perform the actual handling operations. Nodes for point cloud processing were written using the C++ and the PCL library [19] while the node for object detection was written using Python and Keras framework [20]. Once either of the nodes output a grasp pose, it is sent to the robot control node, which implemented a position controller using the ROS Moveit motion planning framework. The planner first checks for any collision areas as custom-defined in the URDF file. It then checks for any singularities and executes the planned path based on the given 3D pose of the blood bag computed in the camera frame and transformed into the robot base frame.

Vacuum Gripper

The gripper used for this experiment was a customized retractable rubber cup vacuum gripper, as shown in Fig. 7. A retractable rubber cup of diameter 5*cm* was attached to the pipe, which in turn was attached to a venturi pump producing the vacuum. The pump produced a negative pressure of approximately 52 *kPa*, which was empirically found to be enough to pick and hold onto the bags for transportation. The retraction part of the cup allowed for the cup to deform

Table 1 Vision system specifications

	RGB camera	Depth camera
Resolution	1920 × 1080	up to 1280 × 720
Frame rate	30 fps	upto 90 fps
FOV	64° × 41° × 77°(±3°)	87° × 57°(±3°)
Technology	Rolling shutter	Active IR stereo

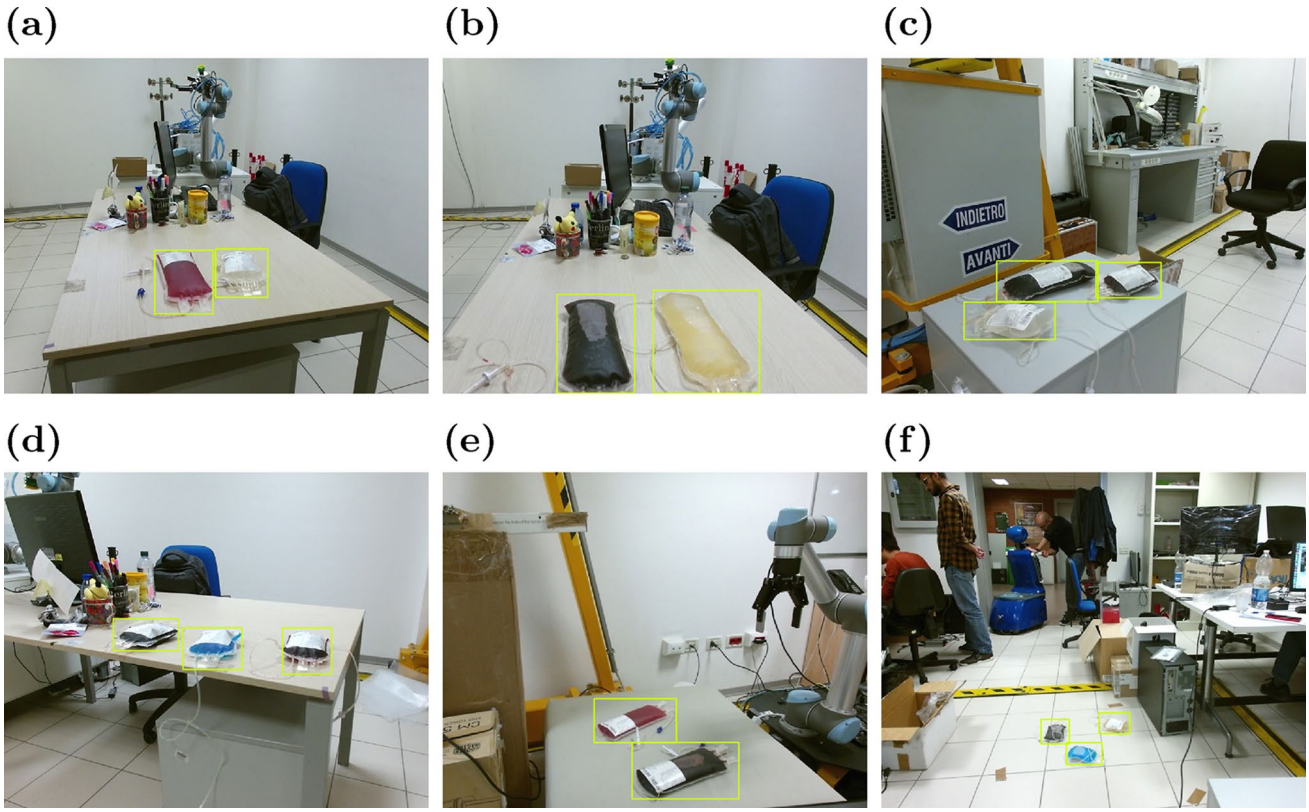

Fig. 8 Example images with annotations in the blood bag data set. **a** Location A—1 small and 1 big bag facing up, **b** Location A—2 big bags facing down, **c** Location B—2 small and 1 big bag facing up, **d** Location A—3 small bags facing up, 2 big bags facing up, **e** Location C—2 big bags facing up, **f** Location D—small bags facing down and 1 small bag facing up

on pressing against the bag's surface which ensured that a tight seal was formed.

Vision System

The acquisition of RBG and depth images of the scene was done using an Intel D435 depth camera (shown in Fig. 1) with the specifications given in Table 1.

Blood Bag Dataset Collection and YOLO Training

Blood Bag Dataset Collection

Two different-sized blood bags of volume 500 cm^3 and 1000 cm^3 each were chosen to construct the training dataset[1]. Ten bags were filled with different colored liquids (red, pink, blue, transparent, and yellow). 500 bag images were taken in six different environmental and lighting conditions, with varying number of bags in each image, as shown in Fig. 8. The bag images were then manually annotated with bounding boxes using the LabelImg tool [21] to obtain ground truth labels in the image coordinates as XML files. To upscale the dataset, the following data augmentation techniques were applied: random brightness change, random rotations about the image center, flipping about both x and y axes, random

[1] **Blood bag dataset**.

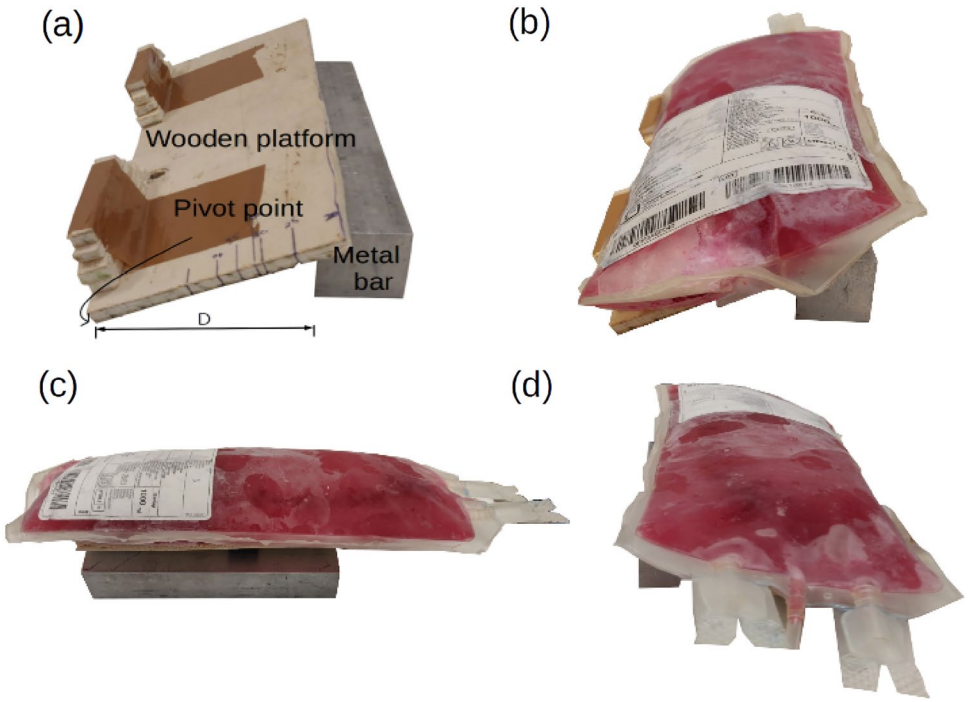

Fig. 9 **a** Wooden platform with adjustable metal bar. **b** Platform with bag at 0°–20° orientation-tilt, **c** Platform with bag at 90° –30° orientation-tilt, **d** Platform with bag at 180°–20° orientation-tilt

translations, jitter noise, and random combinations of these to produce approximately 15000 blood bag images.

YOLO Training

From the collected data, 85% of the images were used for training while the remaining were reserved for validation. Custom anchors box sizes were also computed for our dataset using a *k*-means clustering algorithm on all bounding box label coordinates. The network was trained for 250 epochs on an NVIDIA GeForce 1080 GTX GPU. Adam's optimizer was used with a learning rate of 0.0001. Models were saved every 50 epochs and the lowest validation error model was used for our experiments.

Bag Orientation-Tilt Angles

The experiments were designed such to test the proposed strategies on different bag tilt angles and orientations. First, to change the bag's tilt angle with respect to the camera, two wooden platforms, one for each bag size, were designed. The big platform is shown in Fig. 9a and the same platform with the bag placed is shown in Fig. 9b. These platforms allow us to stick to our assumption about the bags laying on a flat surface. This is because most part of the platform is hidden behind the bag (Fig. 9a–c) while the remaining parts are removed during the clustering and bottom removal phases. The platform can be tilted at different angles by changing the distance **D** of metal bar from the pivot point as shown in Fig. 9a while its orientation is changed by rotating the platform about the axis perpendicular to the table plane. With this arrangement, we tested our strategies at 4 different tilt angles: 0°, 10°, 20° and 30°. Added to this, 4 different bag orientations were chosen: 0°, 90°, 180° and 270°. To achieve a random tilt angle, random objects such as stones were placed under the bags. Finally, pick and place operations were conducted on frozen bags. For each strategy, all 4 orientations were tried 3 times at each tilt angle with an additional of 10 trials of random combinations of orientations-tilt angles. This resulted in 232 trials (4 strategies × 4 orientations × 4 tilt angles × 3 trials + 4 strategies × 10 random orientation-tilt combinations) for each bag size making it a total of 464 trials for both sizes. Three different orientation-tilt combinations are shown in Fig. 9b–d. For each orientation-tilt angle, the gripper pose was computed using the strategy being tested and the vacuum gripper was pushed down by an additional 1 *cm* to ensure a strong seal between the gripper and the bag's surface. A trial was considered a success if a seal was formed between the bag and the vacuum gripper and the arm was able to carry it back to the home position without breaking the seal.

Results and Discussion

YOLO Training

Results for training and validation losses on our dataset for training the YOLO-V3 are shown in Fig. 10. We achieved a

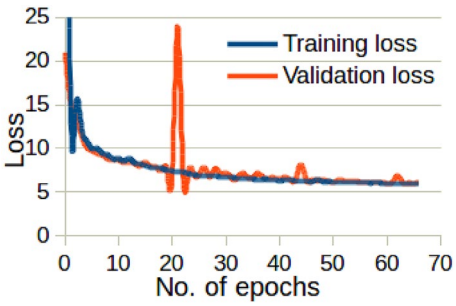

Fig. 10 Training and validation loss on blood bag dataset

bag size-wise analysis is presented in the bar chart given in Fig. 11 and some selected visual results are shown in Fig. 12. While all methods reported average success rates of above 80% as shown in Table 2, a number of problems associated with the failed attempts are identified which explain the different success rates achieved by different methods:

- Grasp attempted at the edge of the bag resulting in an inadequate seal formation as shown in Fig. 12 (row 3 image 4).
- Bags melting resulting in a deformable surface causing the seal to break midway during the pick and place operation.

Table 2 Success rates for all the trials

Method	Tilt angles (degrees)										Average success rate (%)
	0°		10°		20°		30°		Random		
	S	B	S	B	S	B	S	B	S	B	Both sizes for all trials
YG	75.0	75.0	**100.0**	91.7	91.7	75.0	91.6	91.7	80.0	80.0	**85.2**
NE	83.3	91.7	***66.7***	91.7	**100.0**	83.3	83.3	**100.0**	80.0	70.0	**85.0**
ZE	**100.0**	**100.0**	91.7	**100.0**	***66.7***	***66.7***	83.3	***58.3***	70.0	70.0	**80.1**
ASCC	75.0	**100.0**	***66.7***	**100.0**	75.0	**100.0**	83.3	**100.0**	80.0	70.0	**85.0**

NE normal error, *ZE* Z error, *ASCC* average surface curvature change, *YG* YOLO grasp, S: Small bag (500 cm^3), B: Big bag (1000 cm^3)

The maximum score for each strategy is written in bold text while the trials scoring notably low scores are written in bold and italic text

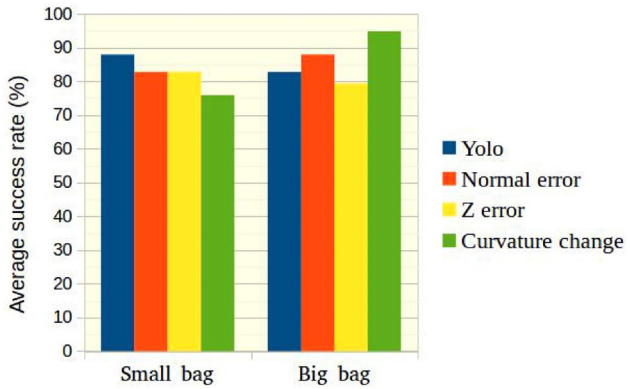

Fig. 11 Average success rate for all methods by bag size

- Bag laying on its curved surface (Fig. 1) and slipping away after the application of pressure from the gripper.
- Noisy depth measurements provided by the camera causing a wrong estimate of the 3D world coordinates of the grasp point.
- The grasp point computed being further away from the bag's center of gravity (COG) resulting in a big moment arm causing the bag to drop.
- The bag's inlet/outlet part satisfying the selection criteria for the point cloud-based strategies resulting in the gripper failing to form a seal at or near this area.

training loss of 5.48 and a validation loss of 6.2 at the end of training for 250 epochs.

Quantitative and Qualitative Analysis

All the methods are evaluated against success rates achieved by each of them. Success rates for each method at different tilt angles for both bag sizes as well as the average success rate for all methods and all trials are reported in Table 2. A

As regards success rates, ***YOLO grasp*** method leads with an average success rate of 85.2 %. The method was observed to maintain consistency in performance across bag sizes as well as tilt angles with success rates of 87.9 % and 82.8 % on small and big bags respectively. It performed better because the grasp point is chosen as the center of the predicted bounding box. As a result, this method doesn't suffer from grasp attempts away from the bag's COG. However, at different tilt angles, the bag's bounding box has a shorter width since the camera is able to view only part of the bag, resulting in the box's center appearing near the edge of the

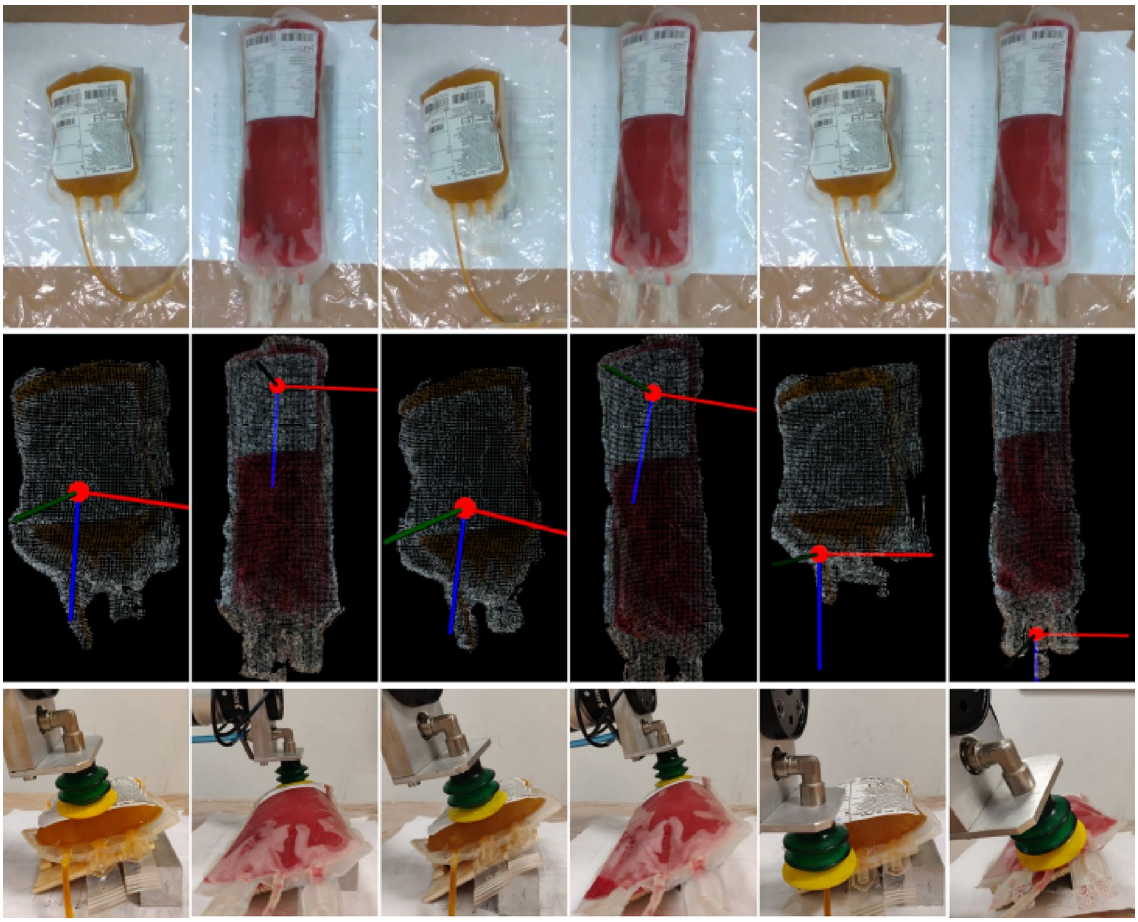

Fig. 12 Different bag orientation-tilt combinations are shown in row 1, their corresponding optimal grasp pose proposals are shown in row 2 and their corresponding real-world grasp attempts that resulted in a success or failure are shown in row 3. Row 1: blood bags at different orientation-tilt angles left to right: 180°–20°, 180°–20°, 180°–30°, 180°–30°, 180°–20°, 180°–30°. Row 2 images 1–4: corresponding optimal grasp poses computed by *ASCC*, image 5: optimal grasp pose computation by *NE*, image 6: optimal grasp pose computation by *ZE*. Row 3 images 1–4: grasp attempts that resulted in success, images 5–6: grasp attempts that resulted in failure

bag. Thus, there are more chances of failure due to a grasp attempted on or near the edge. Another reason limiting this method from achieving an even higher success rate is the method's inability to take irregularities on the bag's surface into account since it always chooses the center of the bounding box as the grasp candidate irrespective of the surface topology. These irregularities, if present near the proposed grasp point, prevent the formation of an adequate seal.

Average surface curvature change with an average success rate of 85.0 % (Table. 2) achieved a performance on par with the *YOLO grasp* yet differs in the causes of failure. First, this method searches for patches on the bag's surface with the lowest value of curvature change. Query patches near the bag's center are naturally bad grasp candidates for this method because the surface starts to curve downwards as we move toward the bag's edge. This results in a big curvature change value for a query patch near the center forcing the method to estimate grasp candidates on the curved surface which is away from the bag's COG. Successful grasps of such type are shown in Fig. 12 (row 3 images 1, 2 and 3). Failure is linked to this computing of grasps away from the bag's center as sometimes the proposed grasp area also includes the bag's edge causing inadequate seal formation as shown in Fig. 12 (row 3 image 4). This behavior was predominantly observed in the case of small bags because they tended to develop larger curvatures (due to the ice expanding more in a smaller volume) on a smaller grasp search area (Fig. 11). *ASCC* significantly improved in performance from achieving a success rate of 75.9 % on the small bag to 94.9 % on the bigger one (Fig. 11). One observation made for this result was the big bag slipping away seldom due to its big size and greater weight which make it more stable as opposed to the small bag which tended to slip away easily both due to its smaller size and weight as well as laying on its curved surface (greater curvature owing to ice expanding more in smaller volume) making it unstable.

Normal error achieved similar performance to that of *ASCC's* with an overall success rate of 85.0 % and success rates of 82.8 % and 87.9 % on small and big bags respectively (Fig. 11). Similar performance meant similar reasons for failure. As this method searches for patches on the bag's surface with points having normal vectors pointing in similar directions, grasp candidate proposals are moved further away from the bag's center because the normal vectors' orientation near the center change to a greater extent due to the surface beginning to curve downwards starting from the center. As a result, most failures result from inadequate seal formation, particularly for the small bag as shown in Fig. 12 (row 3 image 4). Since both the *NE* and the *ASCC* are likely to propose grasp candidates on the bag's top curved surface, the bag is also more likely to slip away on an attempted grasp due to force application from the side if it lays on its bottom curved surface as shown in Fig. 1.

Finally, the ***Z-error*** method achieved the lowest success rate of 80.1 % among the proposed methods. It reported inconsistent performance both across bag sizes and tilt angles. It achieved a success rate of 100.0 % in some cases and simultaneously reported the lowest achieved success rate of 58.3 % in another. This shows that the strategy is less robust to differing conditions. At higher tilt angles the camera only observes a partial and side view of the bag. Since this method searches for patches on the bag's surface which are parallel to the tabletop, it is less likely to find such patches at higher tilts and there is a greater likelihood of failure. Another reason for this method's failed attempts is the interference from the the bag's inlet/outlet part which stays relatively parallel to the table top in different bag orientation-tilt positions. Thus, the method mistakes this area as a good grasp candidate and fails as shown in Fig. 12 (row 3 image 5). These reasons explain the success rates highlighted in red for the *Z-error* in Table 2.

Conclusion

In this paper, four different strategies, both analytical and a hybrid of analytic and deep learning approaches to infer optimal grasp poses for vacuum grippers to automatise pick and place operations of blood bags were presented. Traditionally, vacuum grippers are only used for objects with flat surfaces, but it was successfully shown that these grippers can be adapted to any surface topology. Although the methods focus solely on handling blood bags, the computations involved make it possible to adapt the methods to a variety of different objects after a slight parameter tuning. The methods are also fully capable of proposing grasp candidates in real time.

Although all the methods achieved high success rates under favorable assumptions and a controlled environment, there are still limitations to these methods and require improvements before being adapted to real-world scenarios. In particular, these methods don't take the bag's boundaries into consideration while computing the grasp poses resulting in some grasps attempted on the bag's edges. Another factor limiting the performance of these methods was the noisy point cloud data obtained from the depth camera which sometimes caused inconsistent pose computations. Another shortcoming of the methods was their inability to receive a feedback in the quality of seal formed between the gripper and the bag's surface.

Stemming from these issues future work will focus on computing the bag's boundaries to define a graspable area. Since vision sensors are inherently noisy, the use of CNNs to measure surface normals instead of computing them analytically as suggested in [8] will be investigated. An adaptive mechanism to improve the seal quality based on feedback will be studied. To bypass the analytic calculations required to infer the grasp pose, we will also investigate using vision sensors with Deep reinforcement learning (DQN, DDPG) to learn grasp poses in an end-to-end manner.

Funding Open access funding provided by Università degli Studi di Firenze within the CRUI-CARE Agreement. No funding was received to assist with the preparation of this manuscript.

Availability of data and materials The datasets generated and/or analyzed during the present study are from the corresponding author on reasonable request.

Declarations

Conflict of interest The authors have no conflicts of interest to declare that are relevant to the content of this article.

Ethics approval Not applicable.

Consent to participate Not applicable.

Code availability Not applicable.

References

1. You F, Mende M, Štogl D, Hein B, Kröger T. Model-free grasp planning for configurable vacuum grippers. In: 2018 IEEE/RSJ international conference on intelligent robots and systems (IROS), (2018); pp. 4554–4561. IEEE
2. Correll N, Bekris KE, Berenson D, Brock O, Causo A, Hauser K, Okada K, Rodriguez A, Romano JM, Wurman PR. Lessons from the amazon picking challenge. (2016); CoRR abs/1601.05484. 2016
3. Redmon J, Farhadi A. Yolov3: an incremental improvement. (2018); arXiv preprint arXiv:1804.02767
4. Jiang P, Ishihara Y, Sugiyama N, Oaki J, Tokura S, Sugahara A, Ogawa A. Depth image-based deep learning of grasp planning for textureless planar-faced objects in vision-guided robotic bin-picking. Sensors. 2020;20(3):706.
5. Mahler J, Matl M, Liu X, Li A, Gealy D, Goldberg K. Dex-net 3.0: computing robust vacuum suction grasp targets in point clouds using a new analytic model and deep learning. In: 2018 IEEE international conference on robotics and automation (ICRA), (2018); pp. 5620–5627. IEEE
6. Wang S, Jiang X, Zhao J, Wang X, Zhou W, Liu Y. Vision based picking system for automatic express package dispatching. In: 2019 IEEE international conference on real-time computing and robotics (RCAR), (2019); pp. 797–802. IEEE
7. Sajjan S, Moore, M, Pan M, Nagaraja G, Lee J, Zeng A, Song S. Clear grasp: 3d shape estimation of transparent objects for manipulation. In: 2020 IEEE international conference on robotics and automation (ICRA), (2020); pp. 3634–3642. IEEE
8. Zhang Y, Funkhouser T. Deep depth completion of a single rgb-d image. In: Proceedings of the IEEE conference on computer vision and pattern recognition, (2018); pp. 175–185
9. Shao Q, Hu J. Combining rgb and points to predict grasping region for robotic bin-picking. (2019); arXiv preprint arXiv:1904.07394
10. Ronneberger O, Fischer P, Brox T. U-net: convolutional networks for biomedical image segmentation. In: International conference on medical image computing and computer-assisted intervention, (2015); pp. 234–241. Springer
11. Han M, Pan Z, Xue T, Shao Q, Ma J, Wang W, et al. Object-agnostic suction grasp affordance detection in dense cluster using self-supervised learning. docx. arXiv preprint (2019); arXiv:1906.02995
12. Shukla P, Kumar H, Nandi G. Robotic grasp manipulation using evolutionary computing and deep reinforcement learning. Intel Serv Robot. 2021;14(1):61–77.
13. Jørgensen TB, Jensen SHN, Aanæs H, Hansen NW, Krüger N. An adaptive robotic system for doing pick and place operations with deformable objects. J Intell Robot Syst. 2019;94(1):81–100.
14. Jørgensen TB, Krüger N, Pedersen MM, Hansen NW, Hansen BR. Designing a flexible grasp tool and associated grasping strategies for handling multiple meat products in an industrial setting. Int J Mech Eng Robot Res. 2019;8(2):220–7.
15. Hasegawa S, Wada K, Niitani Y, Okada K, Inaba M. A three-fingered hand with a suction gripping system for picking various objects in cluttered narrow space. In: 2017 IEEE/RSJ international conference on intelligent robots and systems (IROS), (2017); pp. 1164–1171. IEEE
16. Long J, Shelhamer E, Darrell T. Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, (2015); pp. 3431–3440
17. Kim J, Nocentini O, Scafuro M, Limosani R, Manzi A, Dario P, Cavallo F. An innovative automated robotic system based on deep learning approach for recycling objects. In: ICINCO (2), (2019); pp. 613–622
18. Schnabel R, Wahl R, Klein R. Efficient ransac for point-cloud shape detection. In: Sps S, editor. Computer Graphics Forum, vol. 26. Amsterdam: Wiley; 2007. p. 214–26.
19. Rusu RB, Cousins S. 3d is here: Point cloud library (pcl). In: 2011 IEEE international conference on robotics and automation, (2011); pp. 1–4 IEEE
20. Chollet F, et al. keras (2015);
21. Tzutalin: LabelImg. Free Software: MIT License (2015); https://github.com/tzutalin/labelImg