Reconfigurable Grasp Planning Pipeline with Grasp Synthesis and Selection Applied to Picking Operations in Aerospace Factories

João Pedro Carvalho de Souza, Carlos M. Costa, Luís F. Rocha, Rafael Arrais, A. Paulo Moreira, E.J. Solteiro Pires, José Boaventura-Cunha

1. Introduction

Planning and performing a grasp movement is done effortlessly by humans, but for robots, this is a significant challenge. The current established industrial solutions are only capable of dealing with this problem in well-structured and controlled environments. Typically, these solutions resort to techniques that depend on the operators’ expertise, which manually programs the robotic system, or are based on inflexible, application oriented, software tools (e.g., drive through, lead through and offline programming), which do not convey with modern industry paradigms that ultimately seek for new autonomous and efficient techniques to enhance the flexibility of industrial robotic systems.

For decades the study of grasp techniques in complex scenarios has been explored by the scientific community, which led to the appearance of several analytical [1–6] and data-driven [7] approaches aiming for the improvement of production lines, logistics processes, assembling operations, and bin-picking tasks. Despite these significant contributions, the complexity associated with designing a task-oriented analytical method or build a large dataset, required for the training of Machine Learning (ML) systems, limits the effective adoption of these technologies as an efficient, user-friendly, and applicable industrial solution.

In this context, this paper introduces a reconfigurable robot grasping software pipeline. It is based on a sequential architecture to autonomously compute the grasp solution for a robotic arm in an industrial application. This pipeline is built on top of Robot Operating System (ROS) and “GraspIt!” simulator [8], extending the applicability of Simulated Annealing (SA) [9] with a feasible application time.
With this work the authors goal is to deploy to both the industrial and scientific community a software tool capable of automatically generating robot grasp poses over a set of objects. Namely, this paper presents the backbone of the proposed modular and configurable pipeline, where methodologies and tools already consolidated in the scientific community will be further integrated. Furthermore, this tool will serve as the basis for future developments on the robot grasping topic. Practical results are presented considering a real aerospace factory use case, targeting the execution of intralogistics operations by an omnidirectional robot equipped with a robotic arm, i.e., a mobile manipulator (Fig. 1).

Bearing these ideas in mind, this paper is structured as follows: Section 2 discusses the related work. Section 3 presents relevant background on robot based grasp topic. Section 4, presents the proposed grasp planning software backbone. Finally, in Section 5 the experimental results are presented and discussed, followed by the Conclusions and Future Work (Section 6).

2. Related Work

The robotic grasp was firstly investigated by works such as [1–6]. Typically, they explore the stability of multi-fingered grasps considering closure conditions in wrench space analyses. These approaches demonstrated that the computation of valid grasping poses can be complex according to the task demands and mathematical modeling practical assumptions, e.g., the number of fingers, friction or frictionless contacts, object-agnostic or not. Their formulation, however, allowed the research community to derive some definitions used even in ML methodologies. For example, the criteria formulation to define the grasping quality by Ferrari and Canny in [6] and the force and form-closure conditions of Nguyen [1,2].

Nowadays, and driven by the appearance of new processing technologies and computer techniques, a new category of robotic grasping systems has emerged: the experience-based. In this field, several authors investigate the integration of optimization techniques into analytical analyses [9,10] and ML methods to reach grasping results that can be summarized in supervised learning (structured labeled datasets [11–23] or Learning by Demonstration (LbD) [24–27]) and Reinforcement Learning (RL) [28,29].

Focused in object-agnostic grasp, Saxena et al. in [11] used visual object features in the learning task. Geometric properties of superquadrics shape objects and geometric properties associated with gripper structure were adopted by the authors of [13] and [12], respectively. These works demonstrate how complex is the feature extraction procedure during a ML algorithm supervised training, for logistic regression [11], Support Vector Machine (SVM) [13] or Neural Network (NN) [12].

In LbD [25–27] approaches, the feature modeling is not necessary. However, these techniques could limit the robot grasp efficiency since it is based on the teacher agent performance. Another problem is the high practical effort to build a learning dataset, as required in the RL methodologies [28,29].

The Deep Supervised Learning has gain attention from the researching community that, motivated by the impressive Convolutional Neural Network (CNN) results in object classification, investigate this approach into grasp policies. Works like [14–16], consider the approximation of the grasping detection as an object classification problem, achieving interesting grasp detection results. Yet, just a few of them evaluate the suggested approach in a real grasping problem, and in the instances where the authors do, a decrease in the approach performance is verified. Grasping, picking and handling objects by robots could be more complex considering cluttered environments [30], objects with curved [31] or flat [32] surfaces, flexible [33], and composite [34] materials. Mahler et al. [20–23] propose a Grasp Quality CNN (GQ-CNNs) to solve objects-agnostic grasping problem using suction and two-finger gripper. The author also study ambidextrous, and bin-picking policies. The algorithm was trained with a large dataset, called DexNet that include analytical modeling assumptions. Nevertheless, the authors encounter challenges in grasp flexible, porous objects, and with loose packing.

As shown in this section, the complexity of experience-based algorithms derives from: (i) the creation of the database, (ii) the modeling, and the interpretation of the graspable object. The database needs to be relevant to the task at hand, and the object representation must be defined as a 3D model or only with direct sensing data. Besides that, the gripper technology also needs to be evaluated. As related by Birglen et al. in [35], nowadays, several grippers structures are developed by companies led by the challenges of Industry 4.0. The possibilities for algorithms are vast, and their development must comply with the requirements of the application, the capability of generalization, fast decision, and present some tolerance to the inherent sensing errors.

All the related issues, from analytical to ML methods discussed in the previous section, difficult industries to have a suitable tool to generate grasp solutions according to application demands. Even though a completely generic solution is not achieved, the design of a reconfigurable and modular grasp pipeline, using already state-of-art methodologies could increase productivity in factory environments.

3. Background and Notation

Before presenting the proposed grasp planning pipeline, described in Section 4, this section will discuss some background and notation associated with the challenge at hand.

In this first version of the pipeline the authors assume that the object’s shape is known from the beginning, and that only multi-fingered grippers are used. In this context, the following sections, 3.1 and 3.2, summarize the multi-fingered analytical formulation and the SA based grasp algorithm [9], respectively.

3.1. Multi-Fingered Grasp

A multi-fingered grasp is realized over a set of contacts between the active pairs (the workpiece and the gripper). Therefore, the determination of a suitable configuration of independent grasp points is the primary step of the fingered grasp planning.

The wrench vectors describe the forces and moments that influence a rigid body’s dynamic. These vectors can be used to formulate grasp locations, and a wrench vector is presented below:

Fig. 1. Omnidirectional mobile manipulator (left) and bins with objects (right).
where $\mathbf{F}$ and $\mathbf{r}$ are the vector representations of the forces and the moments. The wrench vectors have 3 and 6 Degree of Freedoms (DOFs) in the case of $\mathbb{IR}^3$ and $\mathbb{IR}^6$, respectively.

The contact models can be categorized as frictionless contact, friction contact (also named hard finger contact), and soft contact [36]. The focus of this paper will be the friction contact, since this model is efficient in the picking application addressed in this paper.

The friction contact model considers the mechanical interaction between the active pairs. Therefore, the wrench convex depends on the friction contact forces, described by Coulomb model of friction: Considering the normal force $f_n$ and the tangential force $f_t$, static friction occurs when there is no slipping between the two surfaces of contact, that is when $|f_t| \leq \mu f_n$, where $\mu$ is a positive value representing the static tangential coefficient of friction.

A wrench representation, w.r.t the $i$-th contact point ($c_i$), is defined as follows:

$$
W_{ci} = \begin{bmatrix}
    f_{ci}^T \\
    \mathbf{0}
\end{bmatrix}
$$

$$
\text{where } FC_{ci} = \{ f \in \mathbb{R}^3 : \sqrt{f_{ci}^T f_{ci}} \leq \mu f_n \} \subset \mathbb{R}^3,
$$

Therefore, it is possible to define the matrix that compose the wrench vector:

$$
W_{ci} = BF_{ci}, \quad \mathbf{f}_i \in FC_{ci}
$$

where $B_{ci}$ is the wrench basis matrix with dimension $p \times n$ where $p$ is the DOFs and $n$ the number of independent forces and moments that constitutes $f_i$. The contact model discussed here has as reference frame the one with the origin coincident with the contact point itself. It is more convenient to refer all contacts in a grasp model to a common frame, generally the center of mass of the work piece. Therefore the wrench transformation matrix is defined as follows:

$$
{^oT}w_{ci} = \begin{bmatrix}
    R_{ci} & 0 \\
    \mathbf{0} & R_{ci}
\end{bmatrix} \in \mathbb{R}^3
$$

$$
\text{where } {^oR}_{ci} \text{ and } \mathbf{t}_{ci} \text{ are the rotation and translation matrix of the } i\text{-th contact point (}c_i\text{) w.r.t. object frame (}O\text{). The } \mathbf{t} \text{ is the linear operator representing the cross product } \mathbf{t}_{ci} \times {^oR}_{ci}. \text{ Hence, the contact map } G_i \text{ is defined as follows:}
$$

$$
G_i = {^oT}w_{ci}^{-1} B_{ci}
$$

Note that it describes the direction of each component of the $i$-th applied wrench and defines the constraints of the contact. The grasp map is the matrix with all contact maps that characterize the contact model (it is also named constraint matrix):

$$
G = \begin{bmatrix}
    {^oT}w_{c1} B_{c1}, & \cdots & {^oT}w_{cn} B_{cn}
\end{bmatrix}
$$

$$
\text{Then, including the magnitude of the forces, a workpiece wrench can be written:}
$$

$$
{^oW} = [G_{ci}, \cdots |f_{ci}, \cdots f_{cn}]^T = GF
$$

$$
\text{where: } F \in FC \text{ and } FC = FC_{c1} \times \cdots \times FC_{cn}
$$

The $W$ also defines the GWS (grasp wrench space) of the grasp. It is obtained by means of the $L_\infty$ and $L_1$ norm. The $L_\infty$ defines the GWS ($W_{ci}$) considering the limitation of the maximum allowable normal contact force, while $L_1$ defines the GWS ($W_{co}$) by the sum magnitude of the normal contact forces. The norms operation yields to:

$$
W_{ni} = \text{ConvexHull}\left( \bigcup_{i=1}^{n} \{ w_{pi}, \cdots w_{pi} \} \right)
$$

$$
W_{nco} = \text{ConvexHull}\left( \bigcup_{i=1}^{n} \{ w_{pi}, \cdots w_{pi} \} \right)
$$

$$
\text{where } w_{pi} \in \mathbb{W} \text{ and } \bigcup \text{ is the Minkowski sum. More detail about the norm operation can be verified in [6].}
$$

The concept of grasp closure evaluates the restraining of an object. A common assumption is the force-closure implies an equilibrium, but the inverse does not apply. A grasp has its convex hull defined by the wrenches that constitute the grasp configuration, i.e., the matrix $^oW$. In a force-closure grasp, the convex hull includes the wrench space origin ($O$), see Fig. 3. According to the definition presented in [37], if all wrenches in $^oW$ positively span the entire wrench space, the grasp will be force-closure. Fig. 3 shows a grasp wrench space (GWS) and a convex hull of grasp configuration for force and non-force-closure, for a planar case with a fixed value for the moment ($r$) in the z-axis. Therefore, it is considered $f_n \in \mathbb{R}^3$, $f_n = (f_x, f_y)$, and the resistance to perturbation in both force axes is evaluated.

Since several configurations can reach a force-closure grasp, quality metrics like $\epsilon$-metric evaluate which one is best. The $\epsilon$ is a normalized value that represents the wrench vector’s distance to the origin ($O$), which is the shortest, i.e., the worst wrench vector to support an external perturbation. An efficient grasp, ideally, has $\epsilon = 1$. The left GWS of Fig. 3 elucidates this metric and, the readers are encouraged to a more detailed review of this grasp definition in [6].

3.2. Simulated Annealing Grasp

The SA algorithm [9] integrated in the “GraspIt!” simulator [8] is one of the tools that the grasping pipeline relies on. Since it was used from the perspective of an end user, a brief explanation will be done here, and any further information can be retrieved on the referenced papers.

The SA is a heuristic optimization algorithm based on the cooling of a set of atoms to a minimum state of energy, and it was first introduced...
by [38] in a Statistical Mechanics optimization algorithm application. The “Very Fast Simulated Re-Annealing” was an improvement made by Ingber at [39] and used here. Since it is based on temperature, Ingber proposed that its cooling process decrease as described by Eq. 9
\[ T = T_0 \exp(-k^{1/D}) \]

where \( D \) is the dimensional search space, \( k \) a SA parameter step, and \( T_0 \) is the SA the initial temperature.

Each algorithm iteration generates new state variables following a rule of neighboring. Considering current and a new variable state as \( S_{\text{current}} \) and \( S_{\text{new}} \), this rule yields Eq. 10.
\[ S_{\text{new}} = S_{\text{current}} + T(1-\text{rand}(0,1))(1 + \frac{1}{T})\text{rand}(-1,1) \]

and the probability to change the state between the current and the new one is described by Eq. 11 where \( Q(\cdot) \) represents the objective function of the optimization problem.
\[ \exp(Q(S_{\text{current}}) - Q(S_{\text{new}})) > \text{rand}(0, 1) \]

Regarding the multi-fingered grasp procedure, the objective function to be optimized by SA need to be related to the hand posture \( p \) and the position and orientation of the wrist \( w \) as follows:
\[ F_{\text{fin}} = f(p, w), \quad p \in \mathbb{R}^d, \quad w \in \mathbb{R}^d \]

where \( d \) is the number of intrinsic DOFs of the hand.

As discussed by [9], the hand posture is defined by eigengrasps, a subspace of movement based on how human generate hand postures. The eigengrasps reduces the DOFs of the hand based on how humans select appropriate grasps and hand postures. Studies show that humans simplify, unconsciously, the problem with a pattern in the movement. More information can be verified in [9,40]. The eigengrasp \( \mathbf{e}_i \) is defined by hand, and it is a \( d \)-dimensional direction vector that represents the motion of a group joint space that constitute it. Therefore, a posture can be defined by Eq. 13.
\[ p = p_0 + \sum_{i=1}^{b} a_i \mathbf{e}_i \]

with posture origin defined by \( p_0 \) and \( b \) the total number of eigengrasps. Since it is a linear combination, the parameter array \( a = [a_0, a_1, ..., a_b] \) will be the optimization variable in Eq. 12 together with the \( w \). Therefore, the dimensional search space \( D \) has a reduced length, i.e., \( D = \text{sizeof}(a) + \text{sizeof}(w) \).

The optimization algorithm tries to minimize the linear and angular distance of the Interest Contact Point (ICP) that constitute the Interest Contact Region (ICR) (Fig. 4) adjusting the discussed optimization variables \( a \) and \( w \). The ICR is a contact region model (a predefined group of distributed ICPs) used to calculate the interaction of the algorithm, thus it is possible to create a feasible procedure. Therefore, the objective function to be minimized is described by Eq. 14, where \( N \) is the number of total contacts in ICR, \( \mathbf{n} \) is the surface normal, \( \alpha \) the distance between the ICP and the object \( (i \in N) \). The scalar \( \alpha \) is a range adjustment factor between the distance and the normalized dot product of the second sum part. It is important to note that the mapping between \( F_{\text{fin}} \) and \( Q \) is realized by simulated interaction in the “GraspIt!” [8].

4. Proposed Grasp Planning Pipeline

The developed grasp planning pipeline is divided into two steps: grasp synthesis and grasp selection (Fig. 5). The grasp synthesis is a tool responsible for generating all the grasp poses, and it is based on the “GraspIt!” simulator. More specifically, it creates a set of hypothetical grasp candidates based on the object’s shape. It is an offline step, i.e., it runs outside the robot system in a setup phase. The generated data is then uploaded to the robot system to be used during the grasp selection step. This step is responsible for choosing the best grasp candidate following a set of heuristics and priorities. It is a task-oriented procedure that analyses the environmental and the run-time constraints of the task. The following sections provide a detailed description of the procedure.

4.1. Grasp Synthesis

A summary of the grasp synthesis is presented in Fig. 6. The methodology used in this step resorts to an optimization algorithm called Very Fast Simulated Re-Annealing [41] (see Section 3.2), that was applied to the multi-fingered grasp problem (Section 3.1) proposed by the authors of [9], and currently embedded in the “GraspIt!” simulator. Its controlled stochastic structure reduces the incidence of the local minima problem, as described by [39].

This optimization algorithm is based on the feedback about the physical interaction between the gripper and the object to be grasped, i.e., in every interaction, a computer simulation is performed (see Section 3.2). Thus, the 3D models of the active fingers are needed. For the use case presented in our proposal, we relied on the Robotiq 2F-85, which is an adaptive two-finger gripper with a maximum opening of 85 millimetres. Experimental tests with this gripper show that it can grasp the object models presented in the use case dataset. The 3D models of the gripper (adapted from [42]) are presented in Fig. 7: the left one is used to test and visualize the possible grasp solutions while the second one is effectively used in the optimization step. Both models are modeled with joints movement capabilities. However, only the second one takes into consideration the contact model, i.e., the physical model interaction associated with the visual model. Consequently, the model used in the optimization is simplified, and it reduces the computational complexity and possible collisions between the links of the gripper itself. An arbitrary number of contact points are selected for the contact model, these points constitute the ICR, and they belong to the fingertips (Fig. 8). The number of points that are considered in the ICR affects the algorithm performance: a large set of points increases the reliability of
Fig. 6. Grasp synthesis process flowchart.

Fig. 7. 3D gripper models used (adapted from [42]). Complete model (left) and simplified model (right).

Fig. 8. Possible configuration of ICR. From left to right the number of contact points increases. The model becomes more reliable but convergence of the algorithm is more prone to issues.

Fig. 9. The grasp selection process. The object with the grasp candidates (top left). The best candidate selected (top right). The picking movement (bottom).

Fig. 10. CAD parts model representation of the grasp test cases, namely, bracket (top left), single-side bracket (top right), double-side bracket (middle left), support bracket (middle right), multi-side bracket (bottom left) and reinforced bracket (bottom right).

Fig. 11. Examples of generated grasps for the bracket object.
the grasp procedure, but it could compromise the convergence of the algorithm.

In the present use case, the gripper has only one einfanggrasp (Section 3.2), simplifying the convergence of the SA algorithm. This approach is adequate to model all joints movement as only one group, since the gripper does not have independent fingers and only performs the opening-closing procedure.

To run the grasp synthesis and initialize the process of Fig. 6, a launch file is configured with the object to be grasped, the gripper type, the ε-value (the quality metric used, discussed in Section 3.1) and the maximum iterations thresholds. A fine-tuning of SA parameters is also possible.

The Grasp Viewer interface (seen in Fig. 11) shows the progress of grasp finding for each iteration. At the end of each SA, a set of good grasps is stored and a new simulation is launched until the maximum iterations are reached.

After the grasp synthesis process executed all iterations of the optimization algorithm, the relevance pipeline evaluates all grasp candidates. Therefore, no redundant grasp poses are generated. This pipeline merges grasps that are close to each other by angular and linear distance based on configurable thresholds, also defined in the launch file. In the end, a configuration file (structured in a YAML file) is exported with all the grasp candidates that will be used by the grasp selection procedure.

4.2. Grasp Selection

The grasp selection is a ROS package designed for choosing the best grasp over a set of previously taught grasp poses of an object. This step is an online procedure; i.e., it is a run-time process performed during the robot task execution. Thus, this operation needs to be fast and reliable. The grasps candidates are specified in a YAML configuration file and loaded in the ROS parameter server. These candidates are unique for each object and are generated by the grasp synthesis step. Once the candidates are created and loaded in the parameter server, the grasp selection pipeline estimates the best grasp candidate for allowing the robot to pick the object. The best grasp candidate is chosen according to a cascade of heuristics defined by the user in a YAML file for each object detected. Therefore, the object needs to be identified and localized before the grasp selection process. The heuristics cascade gives a score for each grasp candidate related to a reference frame (such as the gripper). The one with the lower cost is the eligible candidate. It is possible to define a weight for each heuristic in the pipeline based on the importance level of each method in the application. An illustration of the described procedure is presented in Fig. 9.

Below are the set of heuristics supported by the grasp selection pipeline:

- **Joint space filter**: This method is a filter that discards candidates that exceed joint thresholds using the inverse kinematics of the robotic arm. The inverse kinematics of each possible grasp is calculated using the trac_ik library [43]. This avoids the robot to choose a grasp candidate that requires an impossible, impracticable, or dangerous movement for picking the object.
- **Depth distance**: The score is given according to the depth distance
between the reference (gripper) and object coordinate frames.

- **Euclidean distance**: the score is given according to the Euclidean distance (3D distance) between the reference and object frames.
- **Roll distance**: The score is given according to the roll distance of the object with relation to the reference frame (cumulative Euclidean Z-Y-X order).
- **Pitch distance**: The score is given according to the pitch distance of the object with relation to the reference frame (cumulative Euclidean Z-Y-X order).
- **Yaw distance**: The score is given according to the yaw distance of the object with relation to the reference frame (cumulative Euclidean Z-Y-X order).

### 5. Results and Evaluation

The dataset used in the evaluation of the grasp planning pipeline is constituted by a set of objects frequently stored in aerospace automated warehouses and handled by operators. Fig. 10 presents the CAD representations of this test case which were used in the grasp synthesis pipeline with the model of the RobotiQ 2F-85 gripper (Fig. 7).

The grasp synthesis pipeline automatically generated grasp candidates for each object in an offline phase. Fig. 11 to Fig. 16 show some examples of grasp candidates rendered by the grasp viewer package. It is important to note that for all cases, 30 iterations were chosen to generate the grasps. However, for the multi-side bracket, more iterations were necessary (a total of 80, shown in Table 1). This situation happens because of the topology of the part: the multi-side bracket possesses small (the seven lateral lumps) and large (the base) graspable surfaces. The SA tends to converge to the largest region (see bottom left and bottom right grasps hypothesis of Fig. 15), i.e., to the easiest graspable part of the object shape. Therefore, a higher number of iterations may improve the detection of small graspable regions (like the top left and top right grasps hypothesis of Fig. 15) since the initial gripper’s position can avoid the recurrence of the minima local problem.

It is also important to consider the ICR model presented in Section 3.2. As already mentioned in Section 4.1, this affects the algorithm’s performance. ICRs allocated in the inner part of the gripper could lead the algorithm to not converge. In that case, the fingertip’s edge can collide with the object before the ICR reaches the contact, i.e., the ICR does not touch the lumps’ surface in the case of the multi-side bracket. If many contact points are selected, a contact region larger than the actual grasp region could be originated, and this could lead to

### Table 1
Grasp planning qualitative results.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Bracket</th>
<th>Single-side bracket</th>
<th>Double-side bracket</th>
<th>Support bracket</th>
<th>Multi-side bracket</th>
<th>Reinforced bracket</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iterations</td>
<td>30</td>
<td>30</td>
<td>30</td>
<td>30</td>
<td>80</td>
<td>30</td>
</tr>
<tr>
<td>Time of convergence (offline phase)</td>
<td>22min33s</td>
<td>42min16s</td>
<td>48min04s</td>
<td>32min48s</td>
<td>158min54s</td>
<td>39min24s</td>
</tr>
<tr>
<td>Number of force-closure grasps generated</td>
<td>433</td>
<td>278</td>
<td>294</td>
<td>385</td>
<td>1239</td>
<td>272</td>
</tr>
<tr>
<td>(offline phase)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Number of relevant grasps (offline phase)</td>
<td>48</td>
<td>83</td>
<td>42</td>
<td>44</td>
<td>150</td>
<td>37</td>
</tr>
<tr>
<td>Number of grasps after human supervision</td>
<td>38</td>
<td>83</td>
<td>38</td>
<td>44</td>
<td>147</td>
<td>34</td>
</tr>
<tr>
<td>(offline phase)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mean decision time (online phase)</td>
<td>0.70s</td>
<td>1.77s</td>
<td>0.66s</td>
<td>0.74s</td>
<td>2.54s</td>
<td>0.57s</td>
</tr>
</tbody>
</table>

Fig. 17. Full picking process of the support bracket. Scanning position (top left). Initial movement (middle left). Approach orientation (bottom left). Approach movement (top right). Grasp action (middle right). Lifting movement (bottom right).

Fig. 18. Experimental sample of grasps over the objects database.
a situation where not every contact point belongs to the object. Small ICR or ICR located very near the fingertip edge instead can generate stable but unpractical grasps. This can be seen in the bracket (top right grasp of Fig. 11) and support bracket (bottom right grasp of Fig. 14). These cases are unpractical since any object sensing errors or bin disturbance can invalidate the grasp selection.

Therefore, it is important that a human supervisor, with the developed Grasp Viewer package, analyses and selects the grasps hypothesis to be applied in the real bin picking scenario.

The convergence time of the grasp synthesis algorithm is presented in Table 1, including the mean decision time. These performance values were generated with a medium-end computer with 12GB of RAM and 1.80GHz CPU (i7-8550U). It is important to note that for each iteration were generated with a medium-end computer with 12GB of RAM and in Table 1, including the mean decision time. These performance values will be added, such as to score the grasp result according to the approaching and lifting vector (since some grasp points can generate collision situations between the scenario and the object).

Although motivated by intralogistic use cases in aerospace factories, the proposed reconfigurable and modular pipeline was designed to be applicable in different scenarios. Our main objective is to integrate it into our bin-picking solution that is still in development. As future work, we also intend to assess the performance of the proposed pipeline for automatic picking of product on conveyors-belts at fast consumer goods factories and for the manipulation of tools in collaborative robot applications.

Declaration of Competing Interest

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

References
