



RESEARCH ARTICLE

Selective 6D grasping with a collision avoidance system based on point clouds and RGB+D images

Caio Cristiano Barros Viturino¹  and Andre Gustavo Scolari Conceicao² 

¹Postgraduate Program in Electrical Engineering, Federal University of Bahia, Salvador, Bahia, Brazil and ²LaR - Robotics Laboratory, Department of Electrical and Computer Engineering, Federal University of Bahia, Salvador, Bahia, Brazil

Corresponding author: Andre Gustavo Scolari Conceicao; E-mail: andre.gustavo@ufba.br

Received: 28 October 2022; **Revised:** 5 September 2023; **Accepted:** 17 September 2023;

First published online: 18 October 2023

Keywords: grasping; deep learning methods; robotic manipulators; computer vision; manufacturing

Abstract

In recent years, deep learning-based robotic grasping methods have surpassed analytical methods in grasping performance. Despite the results obtained, most of these methods use only planar grasps due to the high computational cost found in 6D grasps. However, planar grasps have spatial limitations that prevent their applicability in complex environments, such as grasping manufactured objects inside 3D printers. Furthermore, some robotic grasping techniques only generate one feasible grasp per object. However, it is necessary to obtain multiple possible grasps per object because not every grasp generated is kinematically feasible for the robot manipulator or does not collide with other close obstacles. Therefore, a new grasping pipeline is proposed to yield 6D grasps and select a specific object in the environment, preventing collisions with obstacles nearby. The grasping trials are performed in an additive manufacturing unit that has a considerable level of complexity due to the high chance of collision. The experimental results prove that it is possible to achieve a considerable success rate in grasping additive manufactured objects. The UR5 robot arm, Intel Realsense D435 camera, and Robotiq 2F-140 gripper are used to validate the proposed method in real experiments.

1. Introduction

The advances in research on autonomous mobile and manipulator robots have been remarkable. The interdisciplinary characteristics of robotics have contributed to its exponential progress in recent years. Artificial intelligence and computer vision have significantly supported the results of recent research overcoming past analytical and empirical methods [1].

Robot manipulators that can autonomously manipulate objects of different geometries in different environments have a wide range of applications such as medicine, manufacturing, retail, service robotics, emergency support, and serving food, among others. However, there are still many issues to be solved until they can safely be applied to perform these activities, including but not limited to the complexity in performing grasping in unknown objects with adversarial geometry, and the collision with the robot workspace [2–4].

Grasping is defined by the gripper pose so that an object can be grasped, meeting several relevant grasping criteria, such as object shape, position, material properties, and mass, given an image as a reference [5]. Robotic grasping is one of the fundamental skills in manipulating an object and is still an open area of research [6, 7]. When applying a robotic grasping technique, it is necessary to get an accurate definition of what is a successful grasp. This definition varies according to the technique used. Besides that, there may be several successful grasp poses in different object regions. Therefore, it is crucial to select the positive grasp that represents the greatest success rate [8, 9].

Robotic grasping involves several areas of robotics, such as perception [10], trajectory planning [11, 12], and control. Consequently, its implementation in practice is a challenge. This challenge becomes

even greater when the robot performs grasping on objects of different geometries with an unlimited amount of positions since it requires a high dexterity [13, 14]. Besides that, grasping has shifted from considering relatively simple, isolated objects to grasping geometrically and visually challenging objects in a cluttered environment [15, 16].

A robot can be programmed manually to perform a specific activity, providing detailed instructions to the control algorithm, known as an analytical or geometric method. Analytical robotic grasping methods are referred to as hand-designing features and have been widely used in the past [17]. These methods require the development of a mathematical grasping model that includes the geometry, kinematics, and dynamics related to the robot and the object, which is not known [18]. In addition, surface properties and friction coefficients are not available in advance [19]. Therefore, these parameters cannot be accounted for unknown objects.

Analytical methods also consider that the position of the object and the contact locations between the end effector and object are entirely known. In some analytical methods, grasping poses are previously calculated using a point cloud registration method, which matches the 3D mesh point cloud and the real object models through geometric similarities [20, 21]. Despite the satisfactory performance in known environments, analytical methods are not feasible in unknown, dynamic, and unstructured environments, considering unknown objects.

Rather than analytical methods, empirical methods have the advantage of removing partially or entirely the need for a complete analytical model [22]. These methods focus on using experience-based techniques employing machine learning. These methods generate grasps candidates through trial and error and classify them by using some metric. Empirical methods usually imply the existence of previous experience of grasping, provided with the aid of heuristics and learning [19]. This training process requires the use of real robots [23], simulations [24], or direct assessment in images [8].

Deep learning techniques have provided a considerable advance in robotic grasping applied to unknown objects. Through these techniques, it is possible to extract features from objects that correspond to a specific grasping pose or a set of grasping poses. The results achieved exceed the analytical and empirical models [3, 5, 8, 15, 25].

It is not proper to directly compare results between robotic grasping experiments due to the extensive grasp detection techniques used [1]. The reason is that each experiment differs from the other by using different types and numbers of objects, physical hardware, robot arms, grippers, and cameras. Therefore, different authors recently published benchmark procedures for analyzing the grasping performance in distinct scenarios [26–28].

In the context of additive manufacturing systems [29, 30], it is necessary to apply a robust robotic grasping technique capable of yielding a diverse set of 6D grasps. This is necessary as some grasps may not be kinematically possible or collides with objects in the robot's volumetric space. Deep learning-based grasps techniques provided a great tool to improve the performance of grasps in unknown objects. However, grasps are usually performed in environment that offer a low risk of collision with objects. Techniques to avoid collisions between the robot's gripper and the environment are still an open area of research [31].

This paper proposes a selective grasping pipeline to generate 6D grasps using an RGB+D sensor avoiding collisions between the robot's gripper and the environment. To avoid collisions with nearby obstacles, a new collision detection system and a heuristic method to filter grasps were developed by using a signed distance method. We limit the application of the collision avoidance algorithm exclusively to the gripper, as, in the particular test case, there is no danger of the robotic arm colliding with the 3D printer. Although the method has been tested in an additive manufacturing unit to pick objects from a 3D printer bed, it is not limited to this application and can be adapted for other environments, such as bin picking. An extended analysis of the grasping performance is given with experimental data. This work builds upon our preliminary work [32], which was only evaluated in a simulated environment [33]. According to the simulator documentation, the simulator's collision checking system may occasionally yield wrong contact points, causing unrealistic reaction forces, vibrations, or instabilities, besides not being available for real implementations.

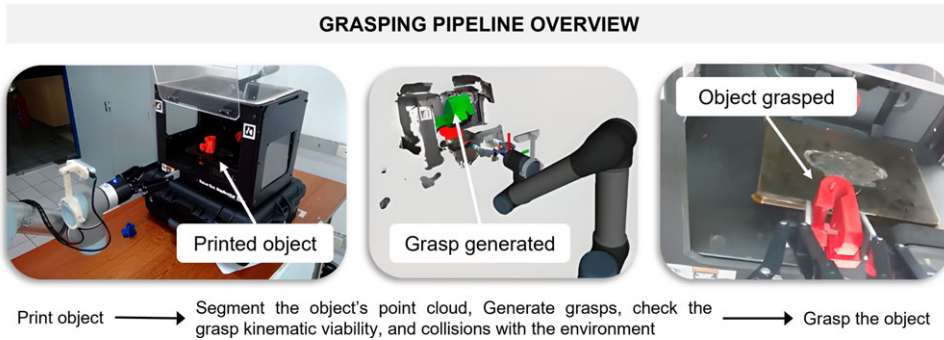


Figure 1. Overview of the proposed grasping pipeline.

An overview of the proposed grasping pipeline can be seen in Fig. 1. This paper has four main contributions: (i) development of a new grasping pipeline to autonomously select and grasp objects of interest placed in a complex environment such as a 3D printer, for additive manufactured objects with low graspability [5]; (ii) development and validation of a low computational complexity collision detection system to discard grasps in collision with the environment and a heuristic method to filter the best grasps; (iii) integration of an object recognition and instance segmentation method, a 6D grasping generator, and the new collision detection system; and (iv) validation of the proposed system using an UR5 Robot Arm Manipulator, a RGB+D camera Intel Realsense D435, and the gripper Robotiq 2F-140. Besides that, an ablation study separating the grasping pipeline stages, an analysis of the relationship between the number of points of the workspace point cloud and the time required to verify collisions, and validation of the deep learning-based object detection and segmentation algorithm for a set of additive manufactured objects confirm the effectiveness of a synthetic dataset generated by a simulator.

The paper is organized as follows. In Section 2, the assumptions are given, and the variables used in this paper are defined. The proposed grasping pipeline implementation is detailed in Section 3. Section 4 presents the approach for object recognition and instance segmentation. The grasping generator and the collision detection using point clouds are presented in Section 5. Section 6 presents the experimental results and an ablation study. Section 7 provides the main conclusions.

2. Problem statement

It has only been in the last few years that convincing experimental data have proven, in practice, the efficiency of grasping methods. Nevertheless, the grasping techniques are often applied for picking objects on planar surfaces such as a table [1] or in a bin [2, 15]. This workspace (table and bin) offers relatively simple test benches to evaluate the grasping performances if compared with constrained spaces such as inside 3D printers. Therefore, 4D grasping methods (also called planar grasps) are enough to generate feasible grasps for planar surfaces but not are suitable for constrained spaces such as inside 3D printers.

To perform grasp in constrained spaces, such as inside 3D printers, it is required to avoid collisions with nearby obstacles such as the printer bed. Therefore, it is necessary to generate a set of feasible grasps with different positions and orientations for the same object, since some grasps are in collision with obstacles in the workspace or kinematically infeasible. The description of the symbols used in this article is as follows:

RGB image. C_i expresses a raw 8-bit RGB image.

Depth image. Let I be an 8-bit 2.5D depth image in which every object in the environment is considered. H and W represent the height and the width of this image, respectively.

Segmentation mask. M_r represents the object segmentation mask.

Object point cloud. N_r evidences the detected object point cloud.

Filtered object point cloud. N_f represents the filtered point cloud of the detected object.

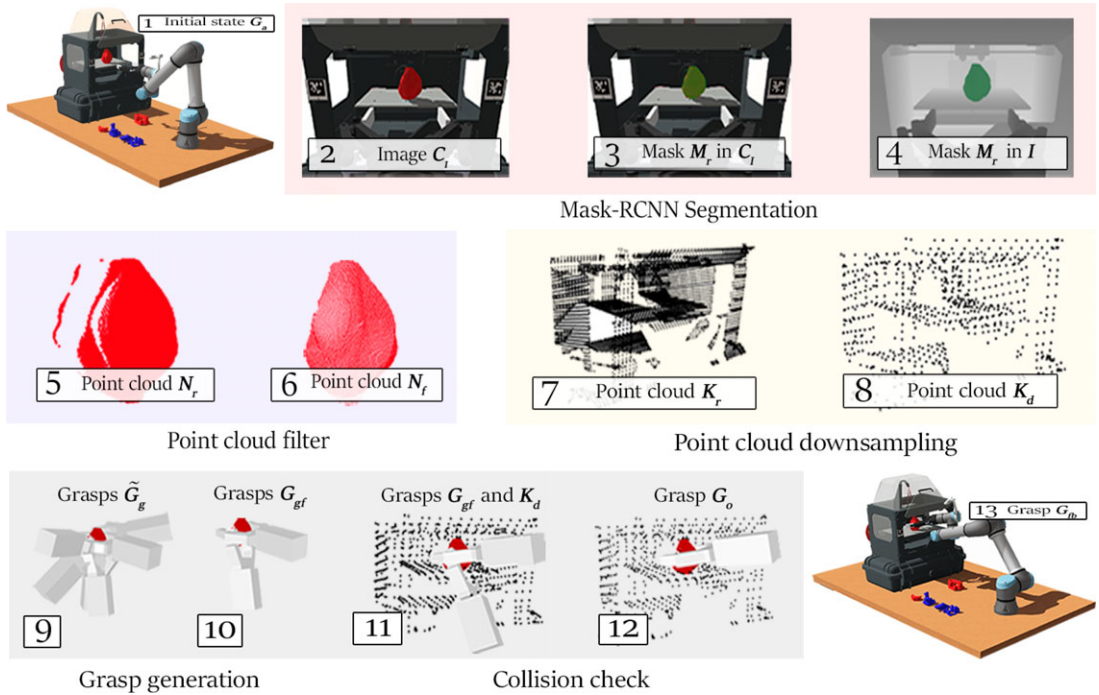


Figure 2. Grasping pipeline.

Printer point cloud. K_r represents the raw point cloud of the 3D printer.

Downsampled printer point cloud. K_d denotes the downsampled point cloud of the 3D printer.

Grasp set. $\tilde{G}_g = (\tilde{P}, \tilde{O})$ denotes a 6D grasps set, in which \tilde{P} and \tilde{O} denote the position and orientation angles, respectively, relative to the camera frame.

Filtered grasps. $G_{gf} = (P_{gf}, O_{gf})$ represents the position P_{gf} and orientation O_{gf} of the filtered grasps by applying the heuristics described in Section 3, relative to the camera frame.

Collision-free grasps. $G_o = (P_o, O_o)$ denote the position P_o and orientation O_o of the collision-free grasps, relative to the robot base coordinate frame.

Grasp on the robot base coordinate frame. $G_{fb} = (P_{fb}, O_{fb})$ represents the position P_{fb} and orientation O_{fb} of the collision-free grasps, relative to the robot base coordinate frame.

Current gripper pose. $G_a = (P_a, O_a)$ describes the actual gripper position P_a and orientation O_a , relative to the robot base coordinate frame.

3. Grasping pipeline

The proposed grasping pipeline is shown in Fig. 2, with the subsequent stages:

1. The initial state of the robot G_a is stored.
2. The image C_i is obtained by positioning the gripper in the front of the 3D printer.
3. The Mask R-CNN receives an image C_i as input and generate a mask M_r .
4. The mask M_r is copied to the depth image I .
5. The pixels of the image I are selected by using the mask M_r . The point cloud N_r is generated by using a backprojection algorithm in the selected pixels of the image I .
6. A statistical outlier removal filter [34] is applied in the point cloud N_r to generate a new point cloud N_f .

7. The printer point cloud \mathbf{K}_r is acquired.
8. \mathbf{K}_r is downsampled to generate a new point cloud \mathbf{K}_d .
9. \mathbf{N}_f is used as input to GraspNet to generate a set of 6D grasps $\tilde{\mathbf{G}}_g$.
10. \mathbf{G}_{gf} is selected from $\tilde{\mathbf{G}}_g$ considering the grasps with a score greater than 80% and \mathbf{O}_{gf} closer to \mathbf{O}_a , so:

$$\mathbf{G}_{gf} = \mathbf{O}_a - \tilde{\mathbf{O}} < \mathbf{O}_m \quad (1)$$

in which \mathbf{G}_{gf} is a set of grasps next to \mathbf{G}_a taking into account a predetermined interval \mathbf{O}_m .

11. Each grasp of \mathbf{G}_{gf} is rejected if any point of the point cloud \mathbf{K}_a lies inside the Robotiq 2F-140 gripper mesh.
12. The grasp without collision \mathbf{G}_o is obtained.
13. The final grasp \mathbf{G}_{fb} is reached by using a quintic polynomial trajectory planning.

Summarizing, after a detection of an object of interest, a pixel-wise segmentation algorithm is applied to create a mask of the objects using RGB images. This mask is used to segment the object in the depth image and then generate a point cloud by using a back-projection algorithm. The point cloud of the object is used to generate a 6D grasp, and the point cloud of the environment is used to check if the generated grasp collides with obstacles. Besides this, each grasp is discarded if it is not kinematic viable.

4. Object recognition and instance segmentation

Mask R-CNN [35] is a deep learning-based object detection and segmentation algorithm. It adds a branch in Faster R-CNN [36] to predict segmentation masks. The mask branch downgrades the performance of the object detection but is still able to reach better performance and accuracy than the COCO instance segmentation task winner in 2016. Mask R-CNN runs at 5 fps on an NVIDIA Tesla M40 GPU.

The layer applied in Faster R-CNN after the Region Proposal Network was used to extract features to classify and apply box regression. It was not designed for pixel-to-pixel alignment between network inputs and outputs. To solve this problem, a quantization-free layer, called RoIAlign, that maintains spatial locations, was applied in ref. [35]. The classes' mask is inferred independently and depends on the RoIAlign to classify and predict categories as is also done on Faster R-CNN.

For each RGB image, Mask R-CNN outputs a segmentation mask, bounding box, and label for each object. Mask R-CNN also requires a backbone architecture. Results show that using a better feature extractor network such as ResNeXt-101-FPN [37] instead of ResNet-50-FPN [38] improves the performance of the Mask R-CNN. In addition, the fully connected layers were removed to build a fully convolutional network. This improved the inference performance of the segmentation task.

4.1. Training and dataset

To detect and segment the object's image, a fine-tune process was applied to the Mask R-CNN. Figure 3 shows the test objects proposed by ref. [5] which were used to evaluate grasping methods due to their low graspability.

A synthetic dataset was generated using the Webots simulator by applying the following pipeline (Fig. 4):

1. An object is randomly positioned in a constrained space and an RGB image is captured by using a virtual camera;
2. The RGB image is generated in simulation;

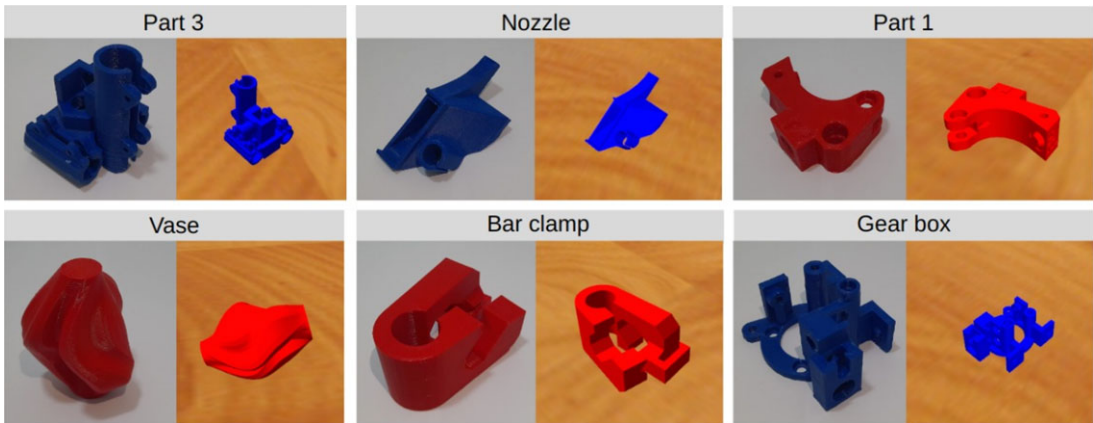


Figure 3. Objects (real and synthetic) used to validate the grasping pipeline.

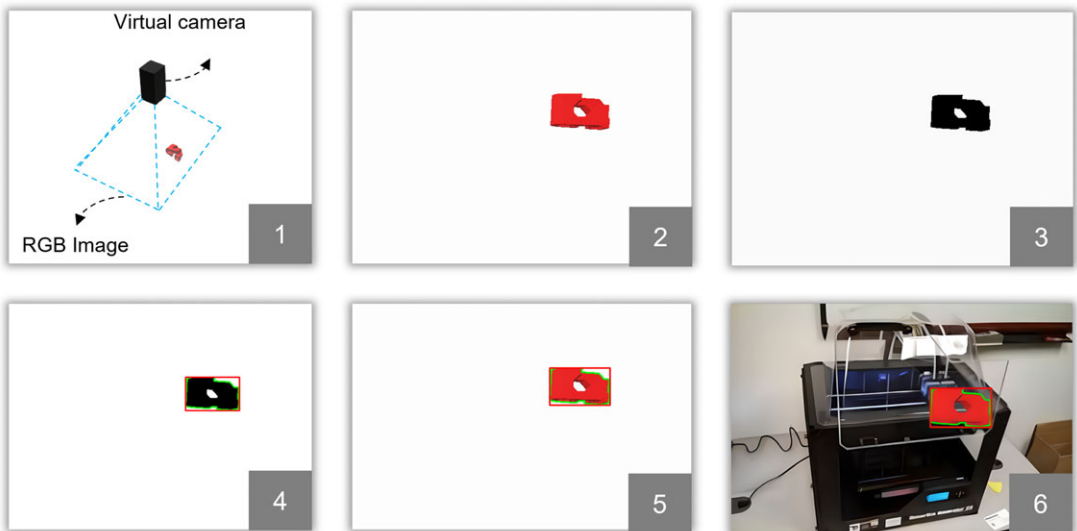


Figure 4. Dataset generation pipeline using the simulator Webots.

3. It was noticed that the object is better segmented in the simulation if the RGB image is turned into a gray image;
4. A bounding box is also automatically generated using the edges of the object's contour extracted from the image;
5. The contour is shown in the RGB image of stage 2;
6. The object pixels in the synthetic RGB image are copied to real images with 3D printers, since the grasps are performed only in this space.

By applying this pipeline, 900 images were created, 150 for each object. The training set and validation set were divided into 80% and 20%, respectively. The average precision (AP), considering the average of IoU thresholds of 0.5:0.05:0.95, was 87.9% for the segmentation task. This AP with averaging IoUs is used to determine the rank of the obstacle detection algorithms of the COCO challenge dataset.

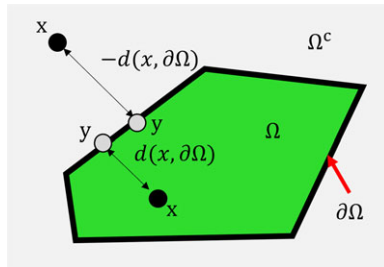


Figure 5. Signed distance used to verify collisions between the point cloud and the Robotiq 2F-140 collision mesh.

The Mask R-CNN was pretrained on the COCO dataset and fine-tuned in 26 epochs. The learning rate was set to 0.0025 with a decay of 10% in epochs 17 and 23. The Stochastic Gradient Descent was used with a weight decay of 0.0001 and momentum of 0.9. The batch size was set to 2.

5. Grasps generation and collision detection using point clouds

GraspNet [3] is a deep learning-based grasping generator [39] consisting of an decoder and encoder trained using Simulated Grasp Trials (SGTs). GraspNet is based on PointNet++ [40] and extracts features from point clouds of the object and gripper for each grasp. This network contains a generator and evaluator module. The generator module takes the latent space samples, the object, and gripper’s point cloud to generate grasps. The evaluation module associates a probability of success for each grasp generated.

It is important to note that the 6D grasping generator was trained using SGTs, considering objects of simple geometry such as bowls, mugs, and boxes. Nevertheless, the objects employed for testing this grasp generator in this work were more complex, although it is still possible to achieve good grasping results as it generalizes well for new objects. Therefore, the 6D grasping generator applied in the grasping pipeline is not optimized to generate grasps for small objects. Consequently, it may take seconds to find a grasp for small objects.

As already mentioned, the 6D grasp generator applied is not capable of analyzing the workspace around the selected object to avoid collisions. To mitigate this problem, a new collision detection system was developed to discard grasps in collision with the environment by using the point cloud of the objects in the workspace.

A simplified mesh of the Robotiq 2F-140 gripper (Fig. 6b) was created to verify collisions with the workspace. For every new grasp generated, it is verified if this mesh collides with the point cloud of any object in the workspace. To check for collisions, it is calculated the signed distance between each point of the workspace’s point cloud and the boundaries of Robotiq 2F-140 collision mesh in the metric space X with a subset Ω , and a metric d , such that

$$f(x) = \begin{cases} d(x, \partial\Omega) & \text{if } x \in \Omega \\ -d(x, \partial\Omega) & \text{if } x \in \Omega^c \end{cases}, \tag{2}$$

$$d(x, \partial\Omega) := \inf_{y \in \partial\Omega} d(x, y). \tag{3}$$

where $\partial\Omega$ denotes the boundary of Ω for any $x \in X$ and \inf denotes the infimum.

Figure 5 exemplifies this statement. Figure 6 shows the workspace used for testing (Fig. 6a), a collision-free grasp (Fig. 6b), and a identified collision between the gripper collision mesh and the workspace point cloud (Fig. 6c).

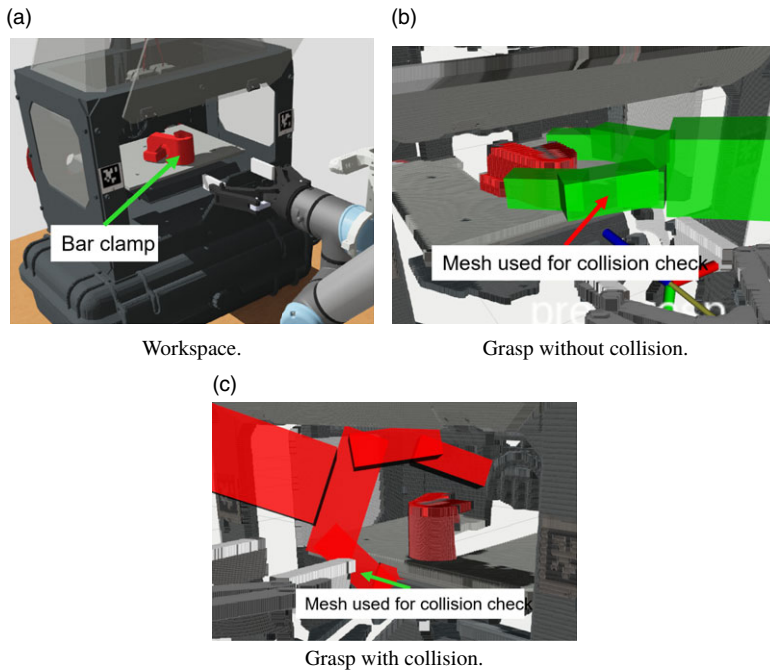


Figure 6. Collision check system in an additive manufacturing system. (a) Workspace configuration used for testing. (b) There are no points inside the gripper collision mesh. (c) There are points of the 3D printer or the object point cloud inside the collision mesh of the gripper.

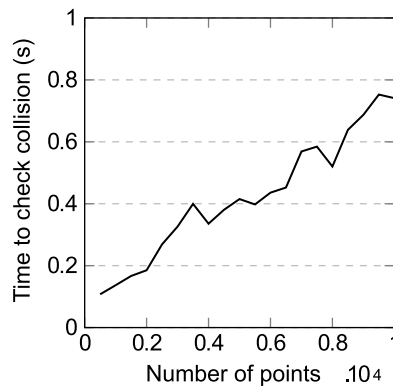


Figure 7. Relationship between the number of points of the workspace point cloud and the time required to verify the collision. It is required 0.8 s to calculate the signed distance for 10,000 points.

The point cloud of the workspace, considering the gripper pose in front of the printer, has on average 10,000 points. It is required 0.8 s to calculate the signed distance for this number of points. It is important to note that the signed distance is calculated again for each grasp generated. As GraspNet generates a diverse set of grasps for each object and most of them are in a collision, it can take considerable time to execute if the point cloud is not downsampled. Figure 7 shows the relationship between the number of points of the point cloud and the time to calculate the collision by using the signed distance.

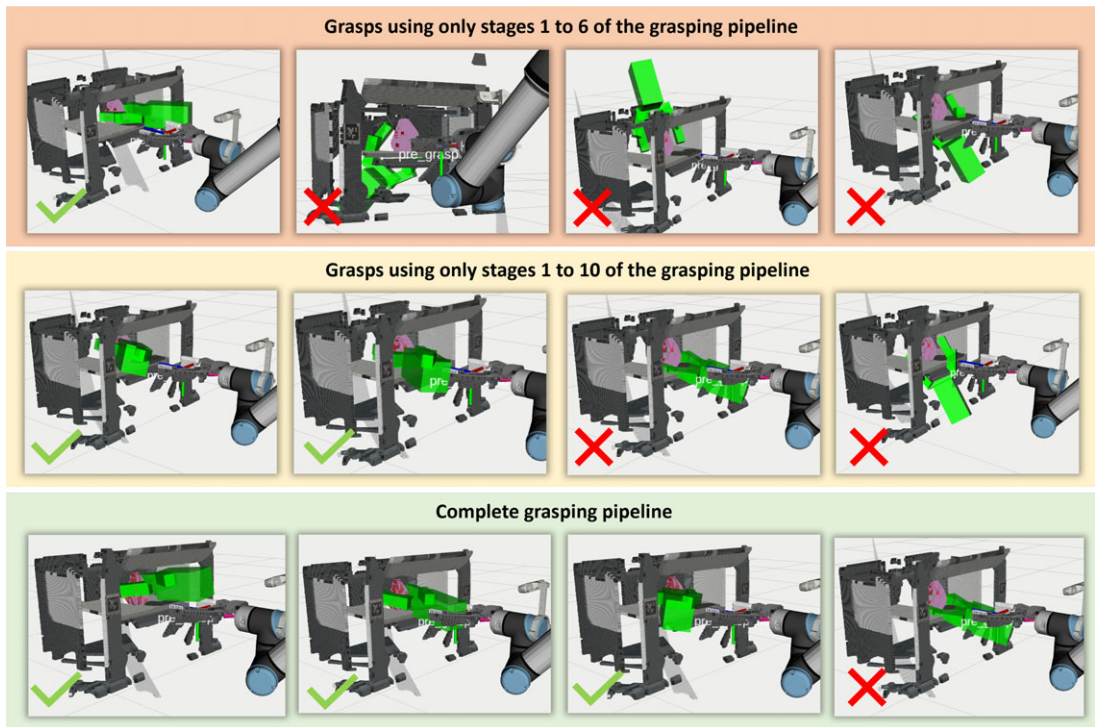


Figure 8. Successful and failed generated grasps generated in each step of the ablation study.

6. Experimental results

This section describes the experiments conducted in an additive manufacturing unit assembled in a laboratory, composed of a 3D printer, an UR5 Robot Arm Manipulator, and a RGB+D camera Intel Realsense D435 mounted on the gripper Robotiq 2F-140. The system was developed using Robot Operating System. The grasping generator and object segmentation algorithms were implemented with the GPU versions of TensorFlow and MxNet, harnessing the power of parallel processing to improve efficiency. The GPU utilized for these computations was the NVIDIA GeForce RTX 3060 graphics card. As a parameter for evaluation, a grasp was considered successful if the object is taken from the 3D printer without slipping through the gripper.

6.1. Ablation study

Some experiments were conducted to better understand the benefits of the integration of the proposed heuristics and the point cloud collision detection into the 6D grasping generator. The objects of Fig. 3 were used in these experiments, and only one object was randomly placed on the 3D printer bed per grasp. In the ablation study, the 6D grasp generator and the instance segmentation technique are employed in each one of the following cases:

1. Using only stages 1 to 6 of the grasping pipeline of Fig. 2. The highest score grasp is chosen in this case;
2. Considering stages 1 to 10 of the grasping pipeline of Fig. 2. A heuristic to filter the generated grasps is applied. The highest score grasp is chosen between the filtered grasps;
3. Employing the complete grasping pipeline of the Fig. 2.

Figure 8 shows instances of successful and failed grasps. Since GraspNet only takes the object point cloud as input without information about the surrounding environment, it generates grasps that often

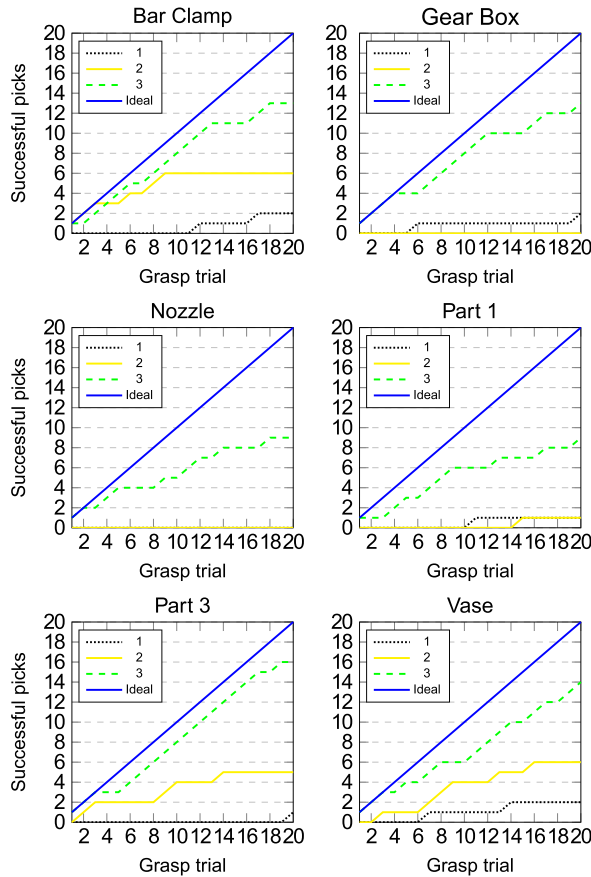


Figure 9. Grasps performed employing the ablation study referred to in Section 6.1. (1) Grasping pipeline from stages 1 to 6, (2) grasping pipeline from stages 1 to 10, and (3) complete grasping pipeline. The objects applied in the experiments are shown in Fig. 3.

collide with the printer. When an orientation constraint is added to GraspNet (stage 10 of the grasping pipeline), the grasp feasibility is improved but it does not guarantee that the robot will not collide with the environment, as clearly seen in Fig. 8. Nevertheless, grasps that are closer to the current end-effector orientation are considered, and grasps far away are ignored. For that, a heuristic already explained in Section 3 is applied. When a collision detection algorithm using the point cloud of the environment is applied, the successful grasps are considerably improved as seen in Fig. 9.

To analyze the performance of the proposed method, 20 pick attempts were performed per object. Figure 9 shows each grasp trial considering the referred ablation study. From the experiments, it can be inferred that the grasp generator does not effectively generate a grasp for small-sized objects such as part 1, nozzle, and gear box. Larger objects such as a vase, part 3, and bar clamp lead to more stable grasps.

In each ablation study, 120 grasp attempts were performed in total. Table I shows the performance obtained for each case of the ablation study. Only eight (or 7 %) grasps were successful when employing the grasping pipeline from stage 1 to 6. When applying the grasping pipeline from stage 1 to 10, 18 (or 15 %) grasps were successful. Considering the entire grasping pipeline, 74 (or 62 %) grasps were successful.

It can be noted through Fig. 10 that the grasp generator and the pixel-wise segmentation methods are fast enough to generate a grasp from 2.6 to 3.4 seconds using the hardware mentioned in Section 6.

Table I. Successful and failed grasping comparison of the ablation study in Fig. 8.

	Grasp success	Grasp fails
Entire grasping pipeline	62%	38%
Stage 1 to 10	15%	85%
Stage 1 to 6	7%	93%

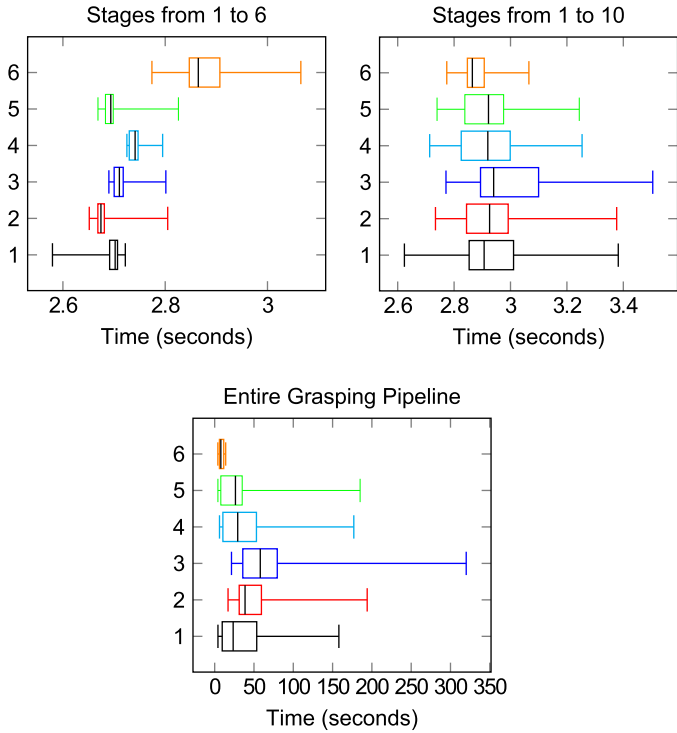


Figure 10. Time to generate a grasp for each object used in the experiments. (1) Bar clamp, (2) gear Box, (3) nozzle, (4) part 1, (5) part 3, and (6) vase. The processing times were based on the ablation study referred in Section 6.1. It is noticed that the entire grasping pipeline is time-consuming due to the collision detection with the point cloud. However, grasp success is considerably increased as shown in Fig. 9.

Despite this lower grasping planning time, the grasping success rate is low. When the grasp collision check using the point cloud is employed, the grasp planning time is increased as well as the success rate as described in Fig. 9.

Despite the success rate of the entire grasping pipeline (62 %), the time consumed to generate a grasp is considerably high, as shown in Fig. 10. Besides that, the time required to generate a grasp highly depends on the object’s geometry. Small objects such as part 1 demand a significant time to find a feasible grasp, and for bigger objects such as the vase, a grasp is generated faster. The reason is that the grasp generator employed was not trained with small objects as seen in ref. [3]. Despite this, Fig. 11 shows that if we set a time threshold to generate a grasp, we would still get a high success rate for some objects. The lower the time threshold is, the lower the success rate because the grasp planner has less time to explore the 6D space.

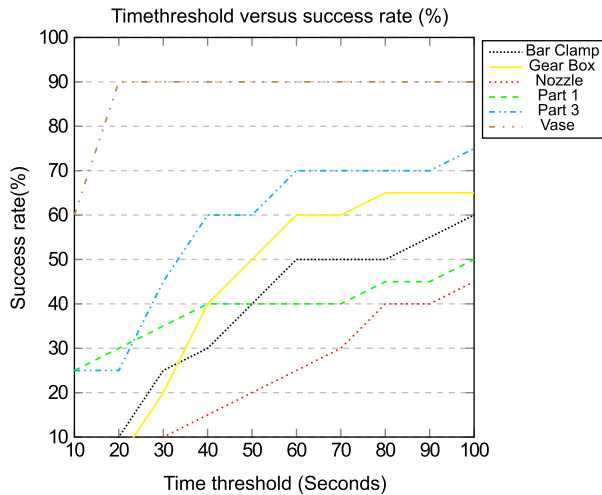


Figure 11. Relationship between time threshold and success rate.

To investigate the grasp repeatability, two poses (position + orientation) were determined empirically for each object, see Fig. 12a, and 15 grasp attempts were performed for each pose for each object, see Fig. 12b. It is important to point out that this repeatability test does not have a statistical value, but an empirical analysis. We can infer that the pose of the object's point cloud N_f in relation to the visual sensor generates different levels of graspability, and consequently the grasping generator produces a set of different grasps, except in cases of objects with a high degree of symmetry (e.g., balls, cubes, rectangles, cylinders, etc).

7. Conclusion

This paper has proposed a selective grasping pipeline to generate 6D grasps. It was accomplished by the integration of an object recognition and instance segmentation method, a 6D grasping generator, and a collision detection system based on point clouds. An extensive analysis of experimental results is provided, involving an ablation study, computational cost for collision detection, and repeatability, applied to additive manufactured objects in a complex environment.

The proposed grasping pipeline generates multiple feasible grasps per object. The variety of grasps produced makes it possible to analyze several viable kinematic solutions and eliminate those that are in collision in the robot's volumetric space, as shown in Fig. 13 and the supplementary material. The main advantage of this solution comes from the integration of important functionalities for grasping systems: (i) selective grasp, the system can grasp and identify the target objects; (ii) segmentation and statistical outlier removal filter to generate object's point cloud in complex environments; and (iii) generation of ranked collision-free grasps. The system with such functionality can be easily adapted to other applications, such as selective pick and place in unstructured environment, selective bin picking, among others.

In future work, we consider a detailed investigation of the grasping efficiency in small printed objects and the object recognition training process improvement. The 6D grasp generator had lower performance when considering small objects. This is even more noticeable when considering a constrained space such as inside the 3D printer. In view of the application mentioned in the paper, we do not perceive a significant issue with allocating additional time to compute the optimal grasp, given that we possess the 3D shape of all printed objects as well as the 3D printers and the grasp computation can occur concurrently with the parts' printing. Nevertheless, incorporating a preprocessing step

(a)

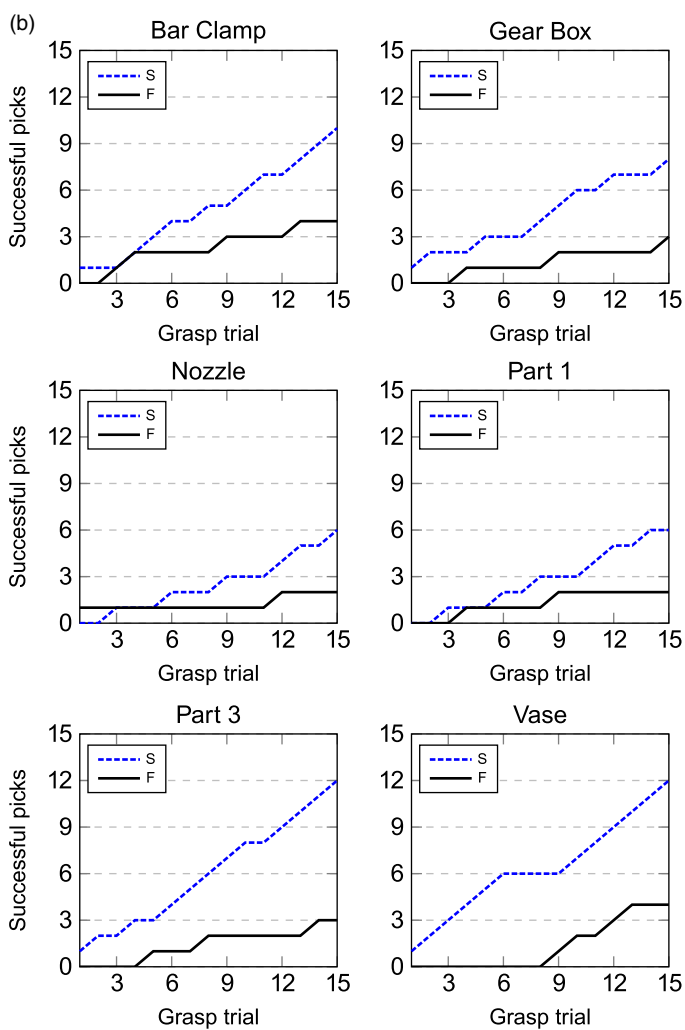
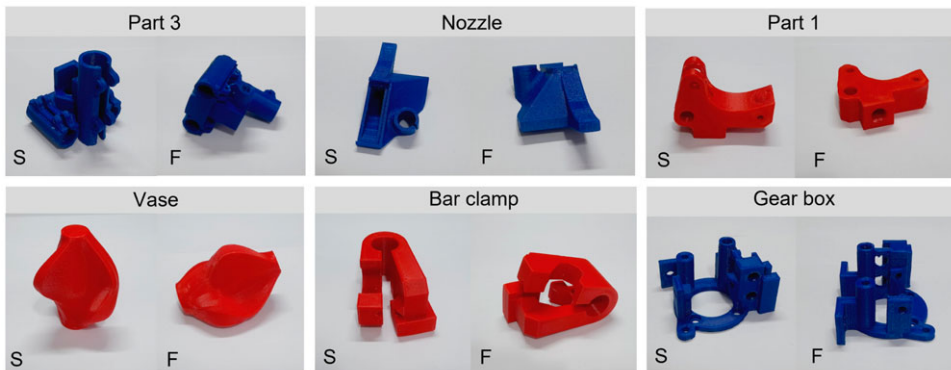


Figure 12. Repeatability test.

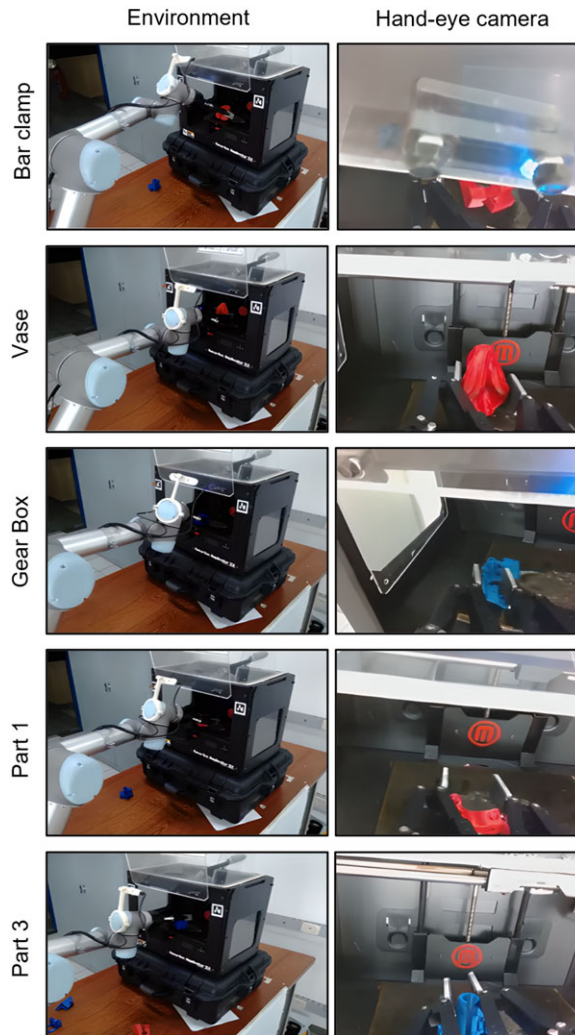


Figure 13. Grasps performed in printed objects inside a 3D printer, a video of these experiments can be found in the supplementary material for this article.

to optimize the voxel representation prior to inputting it into the grasping network is an interesting prospect.

Author contributions. CCBV – discussion, developments, simulation, experiments and algorithm validation, and writing, reviewing, and editing of the manuscript. AGSC – supervision, project administration, discussion, experiments, and reviewing and editing of manuscript.

Financial support. We would like to thank the National Council for Scientific and Technological Development – CNPq grant number [311029/2020-5 and 407163/2022-0], the National Fund for Scientific and Technological Development – FNDCT, the Ministry of Science, Technology and Innovations – MCTI, and the CAPES – Finance Code 001.

Competing interests. The authors declare no competing interests exist.

Ethical approval. Not applicable.

Supplementary material. To view supplementary material for this article, please visit <https://doi.org/10.1017/S0263574723001364>

References

- [1] D. Morrison, P. Corke and J. Leitner, "Closing the loop for robotic grasping: A real-time, generative grasp synthesis approach," *ArXiv Preprint ArXiv:1804.05172* (2018).
- [2] J. Mahler, M. Matl, V. Satish, M. Danielczuk, B. DeRose, S. McKinley and K. Goldberg, "Learning ambidextrous robot grasping policies," *Sci. Robot.* **4**(26), (2019).
- [3] A. Mousavian, C. Eppner and D. Fox, "6-DOF graspnet: Variational grasp generation for object manipulation," **In: Proceedings of the IEEE/CVF International Conference on Computer Vision** (2019) pp. 2901–2910.
- [4] W. Chen, H. Liang, Z. Chen, F. Sun and J. Zhang, "Improving object grasp performance via transformer-based sparse shape completion," *J. Intell. Robot. Syst.* **104**(3), 1–14 (2022).
- [5] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. A. Ojea and K. Goldberg, "Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics," *ArXiv Preprint ArXiv:1703.09312* (2017).
- [6] J. P. C. de Souza, C. M. Costa, L. F. Rocha, R. Arrais, A. P. Moreira, E. S. Pires and J. Boaventura-Cunha, "Reconfigurable grasp planning pipeline with grasp synthesis and selection applied to picking operations in aerospace factories," *Robot. Comput. Integr. Manuf.* **67**, 102032 (2021).
- [7] M. Breyer, J. J. Chung, L. Ott, R. Siegwart and J. Nieto, "Volumetric grasping network: Real-time 6 dof grasp detection in clutter," *ArXiv Preprint ArXiv:2101.01132* (2021).
- [8] I. Lenz, H. Lee and A. Saxena, "Deep learning for detecting robotic grasps," *Int. J. Robot. Res.* **34**(4-5), 705–724 (2015).
- [9] J. Redmon and A. Angelova, "Real-time grasp detection using convolutional neural networks," **In: 2015 IEEE International Conference on Robotics and Automation** (2015) pp. 1316–1322.
- [10] C. R. Steffens, L. R. V. Messias, P. J. L. Drews-Jr and S. S. C. Botelho, "On robustness of robotic and autonomous systems perception," *J. Intell. Robot. Syst.* **101**(3), 1–17 (2021).
- [11] C. C. B. Viturino, U. M. P. Junior, A. G. S. Conceição and L. Schnitman, "Adaptive artificial potential fields with orientation control applied to robotic manipulators," *IFAC-PapersOnLine* **53**(2), 9924–9929 (2020).
- [12] J. Muñoz, B. López, F. Quevedo, R. Barber, S. Garrido and L. Moreno, "Geometrically constrained path planning for robotic grasping with Differential Evolution and Fast Marching Square," *Robotica* **41**(2), 414–432 (2022).
- [13] S. Kumra and C. Kanan, "Robotic grasp detection using deep convolutional neural networks," **In: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems** (2017) pp. 769–776.
- [14] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," **In: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems** (2017) pp. 23–30.
- [15] D. Morrison, P. Corke and J. Leitner, "Multi-view picking: Next-best-view reaching for improved grasping in clutter," **In: International Conference on Robotics and Automation** (2019) pp. 8762–8768.
- [16] D. Kim, A. Li and J. Lee, "Stable robotic grasping of multiple objects using deep neural networks," *Robotica* **39**(4), 735–748 (2021).
- [17] J. Bohg, A. Morales, T. Asfour and D. Kragic, "Data-driven grasp synthesis—a survey," *IEEE Trans. Robot.* **30**(2), 289–309 (2013).
- [18] J. Kober and J. Peters, "Imitation and reinforcement learning," *IEEE Robot. Autom. Mag.* **17**(2), 55–62 (2010).
- [19] J. Maitin-Shepard, M. Cusumano-Towner, J. Lei and P. Abbeel, "Cloth grasp point detection based on multiple-view geometric cues with application to robotic towel folding," **In: IEEE International Conference on Robotics and Automation** (2010) pp. 2308–2315.
- [20] M. Ciocarlie, K. Hsiao, E. G. Jones, S. Chitta, R. B. Rusu and I. A. Şucan, "Towards reliable grasping and manipulation in household environments," **In: Experimental Robotics** (Springer, Cham, 2014), 241–252.
- [21] C. Hernandez, M. Bharatheesha, W. Ko, H. Gaiser, J. Tan, K. van Deurzen, M. de Vries, B. Van Mil, J. van Egmond, R. Burger and others, "Team delft's robot winner of the amazon picking challenge 2016," **In: Robot World Cup** (Springer, Cham, 2016) pp. 613–624.
- [22] G. Konidaris, S. Kuindersma, R. Grupun and A. Barto, "Robot learning from demonstration by constructing skill trees," *Int. J. Robot. Res.* **31**(3), 360–375 (2012).
- [23] S. Levine, P. Pastor, A. Krizhevsky and D. Quillen, "Learning hand-eye coordination for robotic grasping with large-scale data collection," **In: International Symposium on Experimental Robotics** (Springer, Cham, 2016) pp. 173–184.
- [24] D. Kappler, J. Bohg and S. Schaal, "Leveraging big data for grasp planning," **In: IEEE International Conference on Robotics and Automation (ICRA)** (2015) pp. 4304–4311.
- [25] M. Danielczuk, A. Kurenkov, A. Balakrishna, M. Matl, D. Wang, R. Martin-Martin, A. Garg, S. Savarese and K. Goldberg, "Mechanical search: Multi-step retrieval of a target object occluded by clutter," **In: IEEE International Conference on Robotics and Automation (ICRA)** (2019) pp. 1614–1621.
- [26] H. Mnyusiwalla, P. Triantafyllou, P. Sotiropoulos, M. A. Roa, W. Friedl, A. M. Sundaram, D. Russell and G. Deacon, "A bin-picking benchmark for systematic evaluation of robotic pick-and-place systems," *IEEE Robot. Autom. Lett.* **5**(2), 1389–1396 (2020).
- [27] Y. Bekiroglu, N. Marturi, M. A. Roa, K. J. M. Adjigble, T. Pardi, C. Grimm, R. Balasubramanian, K. Hang and R. Stolkin, "Benchmarking protocol for grasp planning algorithms," *IEEE Robot. Autom. Lett.* **5**(2), 315–322 (2019).

- [28] J. J. van Vuuren, L. Tang, I. Al-Bahadly and K. M. Arif, “A benchmarking platform for learning-based grasp synthesis methodologies,” *J. Intell. Robot. Syst.* **102**(3), 1–16 (2021).
- [29] F. S. Costa, S. M. Nassar, S. Gusmeroli, R. Schultz, A. G. S. Conceição, M. Xavier, F. Hessel and M. A. R. Dantas, “FASTEN IIoT: An open real-time platform for vertical, horizontal and end-to-end integration,” *Sensors* **20**(19), 5499 (2020). doi: [10.3390/s20195499](https://doi.org/10.3390/s20195499).
- [30] R. Arrais, G. Veiga, T. T. Ribeiro, D. Oliveira, R. Fernandes, A. G. S. Conceição and P. C. M. A. Farias, “Application of the open scalable production system to machine tending of additive manufacturing operations by a mobile manipulator,” **In: Progress in Artificial Intelligence. EPIA 2019. Lecture Notes in Computer Science**, vol. **11805** (Springer, Cham, 2019) pp. 345–356. doi: [10.1007/978-3-030-30244-3_29](https://doi.org/10.1007/978-3-030-30244-3_29).
- [31] L. Roveda, M. Maroni, L. Mazzuchelli, L. Praolini, A. A. Shahid, G. Bucca and D. Piga, “Robot end-effector mounted camera pose optimization in object detection-based tasks,” *J. Intell. Robot. Syst.* **104**(1), 1–21 (2022).
- [32] C. C. B. Vitorino, D. M. de Oliveira, A. G. S. Conceição and U. Junior, “6D robotic grasping system using convolutional neural networks and adaptive artificial potential fields with orientation control,” **In: Latin American Robotics Symposium (LARS), 2021 Brazilian Symposium on Robotics (SBR), and Workshop on Robotics in Education (WRE)** (2021) pp. 144–149.
- [33] O. Michel, “Webots: Professional mobile robot simulation,” *J. Adv. Robot. Syst.* **1**(1), 39–42 (2004).
- [34] Q.-Y. Zhou, J. Park and V. Koltun, “Open3D: A modern library for 3D data processing,” *ArXiv Preprint ArXiv:1801.09847* (2018).
- [35] K. He, G. Gkioxari, P. Dollár and R. Girshick, “Mask R-CNN,” **In: Proceedings of the IEEE International Conference on Computer Vision** (2017) pp. 2961–2969.
- [36] S. Ren, K. He, R. B. Girshick and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” *CoRR. abs/1506.01497* (2015).
- [37] S. Xie, R. Girshick, P. Dollár, Z. Tu and K. He, “Aggregated residual transformations for deep neural networks,” **In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition** (2017) pp. 1492–1500.
- [38] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan and S. Belongie, “Feature pyramid networks for object detection,” **In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition** (2017) pp. 2117–2125.
- [39] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” *ArXiv Preprint ArXiv:1312.6114* (2013).
- [40] C. R. Qi, L. Yi, H. Su and L. J. Guibas, “Pointnet++: Deep hierarchical feature learning on point sets in a metric space,” *ArXiv Preprint ArXiv:1706.02413* (2017).