2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Environment-Aware Grasp Strategy Planning in Clutter for a Variable Stiffness Hand

Ashok M. Sundaram[1], Werner Friedl[1], Máximo A. Roa[1]

Abstract— **This paper deals with the problem of planning grasp strategies on constrained and cluttered scenarios. The planner sequences the objects for grasping by considering multiple factors: (i) possible environmental constraints that can be exploited to grasp an object, (ii) object neighborhood, (iii) capability of the arm, and (iv) confidence score of the vision algorithm. To successfully exploit the environmental constraints, this work uses the CLASH hand, a compliant hand that can vary its passive stiffness. The hand can be softened such that it can comply with the object shape, or it can be stiffened to pierce between the objects in clutter. A stiffness decision tree is introduced to choose the best stiffness setting for each particular scenario. In highly cluttered scenarios, a finger position planner is used to find a suitable orientation for the hand such that the fingers can slide in the free regions around the object. Thus, the grasp strategy planner predicts not only the sequence in which the objects can be grasped, but also the required stiffness of the end effector, and the appropriate positions for the fingers around the object. Different experiments are carried out in the context of grocery handling to test the performance of the planner in scenarios that require different grasping strategies.**

I. Introduction

Humans have the ability to interact with complex environments and grasp from clutter with ease, but a similar performance is difficult to achieve with robotic manipulators. Robotic grasping from cluttered or restricted environments (e.g. a bin or a shelf with small compartments) poses several challenges, including: (i) objects are enclosed within a narrow space, (ii) objects are placed close to each other, and (iii) the hand has little room to maneuver and grasp the objects.

Typical approaches for robotic grasping rely on the synthesis and evaluation of grasps for a given object and hand. Two main categories of grasp synthesis algorithms exist [1]: analytical approaches, based on geometric or kinematic formulations, and empirical or data-driven approaches that rely on data sampling and/or human observations. To rank multiple grasp options and choose the best possible grasp for a given object, multiple quality metrics have been proposed, with the largest minimum resisted wrench being the most common one [2]. Most of these traditional approaches consider only the object and hand in isolation, and their direct applicability to situations such as grasping in clutter is not feasible. Additional planning is required to consider

*This work was supported in part by the Bavarian Ministry of Economic Affairs, Regional Development and Energy, within the project SMiLE2gether (LABAY102).

[1]All the authors are with the Institute of Robotics and Mechatronics, German Aerospace Center (DLR), 82234 Wessling, Germany. Contact: ashok.meenakshisundaram@dlr.de.

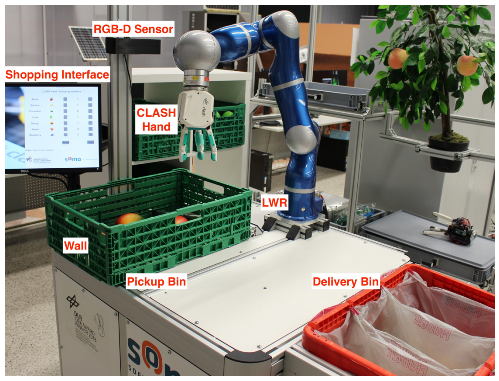


Fig. 1: Overview of the experimental setup used to evaluate the grasp strategy planner. The different hardware components are labeled in red.

other important factors such as which objects are graspable from a bin or shelf, which objects are hindering the grasp process, or what pre-manipulation actions are possible within the constrained narrow space of the bin. We refer to a planner of this kind as a grasp strategy planner.

Grasping from clutter has received increased attention lately, especially using learning-based approaches [3], [4], [5]. While these algorithms have proved applicable to grasp random objects to clear a pile or empty a bin, they lack for strategies to grasp specific objects of interest in the clutter. Moreover, they are mostly restricted to planar (top) grasps, which might not be applicable to generic scenarios. Exploitation of and interaction with the environment or surrounding objects is in general not considered so far. For instance, a book lying on its cover and having a width great than the gripper opening is not suitable for a planar grasp, but it is possible to pierce along the side of the book by restricting its movement with the help of other objects and grasp it, since the thickness of the book is smaller than the gripper opening. Such pre-grasp interactions are important but often neglected within the learning-based approaches. The clutter can be rearranged using push actions in order to grasp a desired object, as presented in [6]. In [7], the robot grasps from clutter while simultaneously moving away objects in the path toward the desired object. Explicitly sequencing the grasping order is rarely considered. The dexterity of the arm in the grasping region is also neglected in the planning process. Planning the grasping order in clutter is tackled in [8] for unknown objects, therefore the properties of the object are not considered. The sequence of grasping is important to de-clutter the bin efficiently and therefore increase the overall success of the system.

While traditional grasping frameworks (including learning-based approaches) for clutter try to avoid contacts with the environment or surrounding objects and aim directly to place the fingers around the object [9], an analysis of human grasping demonstrates that we extensively rely on contacts with the environment for most grasping tasks. A taxonomy of how humans make use of the environment and how this is relevant for robotic grasping is presented in [10]. The environment can be described in terms of environmental constraints (ECs), e.g. elements like a table top, wall, edge, or corner that can be used to successfully grasp objects that would be very difficult to grasp otherwise (e.g. a thin ticket or piece of paper). In [11], grasp planning strategies to use the environmental constraints are discussed and demonstrated by grasping an object in different scenarios using various ECs. Comparing grasps that exploit different ECs and deciding among them which one is optimal for the current scenario has not been considered so far.

The active exploitation of contacts with the environment requires also a different paradigm for the design of robotic hands. Most robotic hands have been designed up to now using rigid links and connections; a detailed review can be found in [12]. Soft or variable stiffness hands have inherent passive compliance and can easily adapt to the environment and absorb unexpected collisions [13]. The recently developed DLR CLASH hand (Compliant Low Cost Antagonistic Servo Hand) [14], whose main feature is its variable stiffness, allows withstanding collisions with the environment and also to change the passive stiffness to adapt to the object or perform pre-manipulation actions. The variable stiffness of the hand can be exploited for a wide spectrum of tasks, from grasping highly fragile objects such as fruits and vegetables, to grasping rigid and heavy objects.

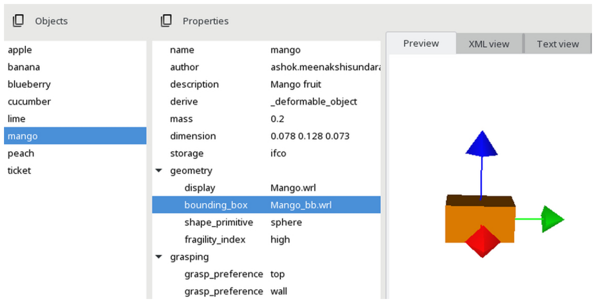
This paper proposes a grasp strategy planner that exploits environmental constraints with a variable stiffness hand for grasping specific objects in cluttered or restricted scenarios. To solve this problem, this paper provides:

- a generic grasp-centric world model that analyzes and lists all objects with their possible grasps, ECs, and pose constraints for each EC
- a grasp sequence planner that returns the best object and EC grasp to execute for the current restricted scenario
- a finger stiffness decision tree for the pre-grasp interaction that considers the object fragility and position uncertainty
- a local finger position planner for highly cluttered scenarios to increase the grasp success probability

Section II describes the various components that form the grasp strategy planner. The hardware setup, other supporting software, and the system integration are discussed in Section III. In Section IV the results of different experiments that show the capabilities of the grasp strategy planner and the complete system are presented. Finally, Section V includes the concluding remarks and future work.

II. Grasp Strategy Planner

The grasp strategy planner comprises three sub-planners: (i) grasp sequence planner, which returns the order in which the objects should be grasped for a better success rate, (ii) finger stiffness planner, which returns the appropriate finger stiffness for the pre-grasp interaction with the object, (iii) finger position planner, which returns the appropriate hand rotation such that the fingers have free regions around the object to slide during pre-grasp motions. A grasp-centric world model coordinates these components and keeps track of the possible actions for the current scenario. The components are detailed in the following sub-sections.



apple
banana
blueberry
cucumber
lime
mango
peach
ticket

name	mango
author	ashok.meenakshisundara
description	Mango fruit
derive	_deformable_object
mass	0.2
dimension	0.078 0.128 0.073
storage	ifco
geometry	
display	Mango.wrl
bounding_box	Mango_bb.wrl
shape_primitive	sphere
fragility_index	high
grasping	
grasp_preference	top
grasp_preference	wall

Fig. 2: GUI interface for the object database, showing details for one object.

A. Grasp-Centric World Model

The grasp-centric world model is the foundation for the grasp strategy planning framework, and describes the current environment and the set of relations and actions applicable to the scenario. The root node of the graph is the robot, which has an associated reachability/capability map [15]. The first layer of nodes attached to the robot node represent the objects in the scenario, which are connected to the robot node via a relation that describes their relative pose. A suitable vision algorithm is used to localize all objects. Various other nodes can be attached to the *object nodes*, including *neighbor nodes* representing neighboring objects, *pose nodes* representing required poses for pre-manipulating the object, and *grasp nodes* representing different grasping actions applicable to the associated object. The different nodes and relationships and the properties they hold are listed in Table I.

TABLE I: Elements of the grasp-centric world model.

Type	Name	Properties	Representation
Root Node	Robot	Capability map	🔴
Node	Object	Pose, Fragility index, Vision score	🔵
	Pose	Reachability index	🟡
	Grasp	Grasp pose, Pre-grasp pose, Fragility index, Vision score, Capability index, MPV magnitude	🟢
Relationship	Pose	Pose	→
	Neighbor	MPV magnitude	
	Reachable	Reachability index	

The object node in the world model saves the following properties: object's relative pose with respect to the robot, its fragility index, related to the maximum force that can be applied on the object, and its vision score, which is a confidence score given by the vision algorithm used to detect the object. The object-specific properties are retrieved from an underlying object database [16]. This database includes information on the object dimensions, fragility, shape primitive, and applicable grasp types. The GUI interface of the object database is shown in Fig. 2.

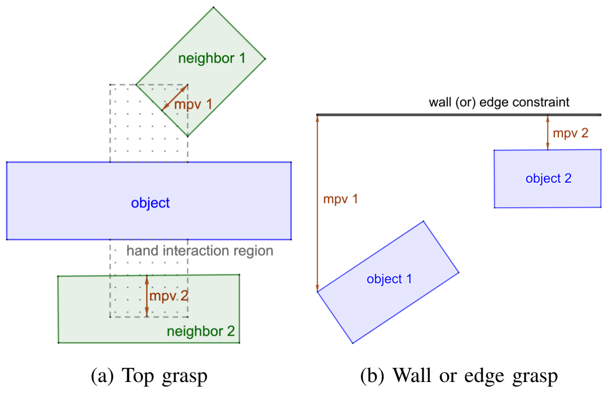


Fig. 3: Minimum push vector for different environmental constraints.

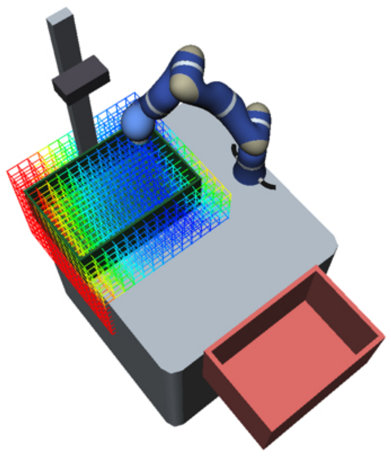


Fig. 4: Capability of the robot arm in the grasping region. The colors correspond to the HSV scale, with blue and red being the highest and lowest local dexterity, respectively.

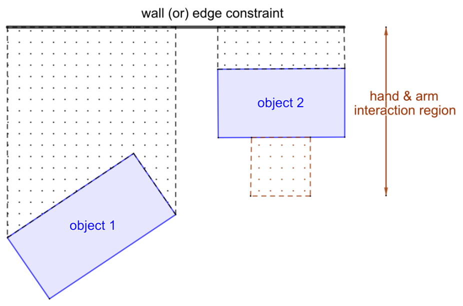


Fig. 5: Collision-free region required to perform wall or edge grasp.

To determine the neighbors of an object, a 2D collision check between the *hand interaction region* of the object and the other objects or environmental constraints (e.g. a wall) in the bin is performed. The hand interaction region is defined around the object and depends on the robotic hand; it accounts for the space the hand needs around the object to interact, for instance to grasp or move the object to an EC. For grasping with the 3-fingered CLASH hand, the length of the hand interaction region is the distance between the thumb and the opposing fingers in a predefined pre-grasp configuration. The width of the region is the distance between the two opposing fingers. All the colliding objects and ECs are added in the world model as children of the object with neighbor relationship. We also compute the *minimum push vector* (MPV), which indicates the distance a neighbor object has to be moved to be able to perform a collision-free interaction with the desired object (Fig. 3a).

An object node also has associated grasp nodes. For each object, the object database provides a list of possible grasp types useful for grasping that object. A grasp node is attached to the object node only if it is feasible in the current scenario. For a top grasp, for instance, the grasp node is added only if it is reachable by the robot. The reachability is evaluated using a capability map [15], which is computed offline for a given robot kinematics. For a given hand pose, a query of the reachability map indicates if the pose is reachable or not. If the top grasp node is added, it also stores the capability index associated with the grasp pose; this index is an indirect measure of local dexterity. The capability of the robot arm in the bin region for our evaluation setup is shown in Fig. 4, and it clearly indicates that some regions of the bin have better dexterity compared to others. More details on reachability and capability analysis can be found in [15].

The top grasp node also stores information related to how cluttered the environment is around the given object; for this, the MPV magnitudes of all the neighbors are added and this total MPV score is also saved as a property of the grasp node. Other properties stored for all grasp nodes include the required grasp pose, pre-grasp pose, vision score, and object fragility index.

For a wall or edge grasp type (EC-grasp), it is important to know how far is the required EC (wall or edge) from the object. The MPV in this case is calculated as the distance between the object and the EC (depicted in Fig. 3b), and it corresponds to the distance the object must be moved to perform the grasp. This MPV along with the neighborhood MPV is stored for the corresponding grasp node. For these ECs, the feasibility check has some additional conditions. First, the space from the object toward the EC is required to be collision-free. Additionally, the wall grasp strategy needs an interaction region for the arm/hand within the bin to perform the grasp, which should also be collision-free. A pictorial representation of both cases is shown in Fig. 5.

In order to check reachability for an EC-grasp type, the path toward the EC is divided into equal intervals ($5cm$ is used in this work). If one of the poses in the path is not reachable, or if the path toward the EC is not collision-free, or if the arm/hand interaction region for the grasp is not collision-free, then the EC-grasp is considered not feasible and the grasp node for that grasp type is not added for the object in the world model. If the EC-grasp is feasible and the grasp node is added, the associated capability index for the node is the average capability index of all the intermediate poses and the EC-grasp pose.

A typical example for the world model is shown in Fig. 6. This world model corresponds to the experiment discussed in Section IV-A.1; the position of the objects in the bin is shown in Fig. 9a. Since all the objects in the bin need to be grasped in this case, all objects are included in the

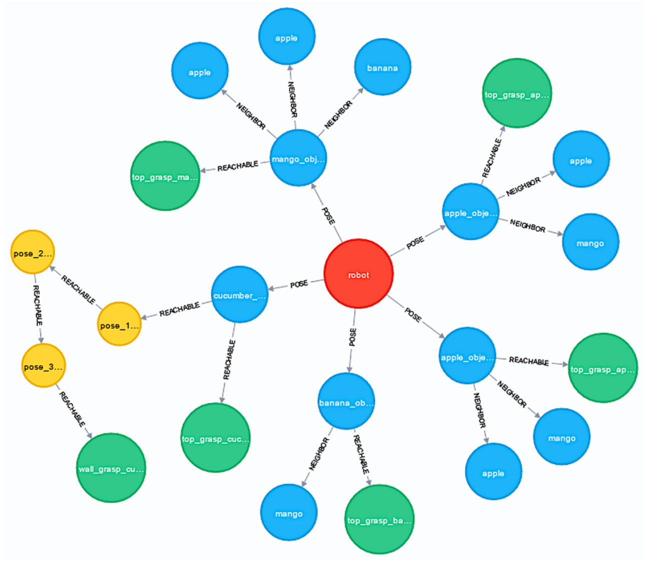


Fig. 6: World model created during the top grasp experiment (Section IV-A.1). The elements of the world model are listed in Table I.

world model. The neighbors for each object are also added. A wall grasp node is created only for cucumber, as it is the only object that satisfies all the feasibilities discussed above. The intermediate poses to move the object toward the wall constraint can also be seen in the model, connecting the object node (cucumber) to the wall grasp node. It must be noted that only one wall of the bin is designated as a wall constraint suitable for grasping (highlighted in Fig. 1) in order to demonstrate the features of the planner in a simple way. However, all four walls of the bin are considered for collision checks to determine object neighbors.

B. Grasp Sequence Planner

The grasp sequence planner calculates a grasp score and sorts all the grasp nodes in the world model. The grasp with the highest score corresponds to the best grasp (higher success probability) to be executed first. The algorithm for this planner is listed in Alg. 1. The grasp score is calculated based on three different scores embedded in the individual grasp nodes: (i) magnitude of the MPV, (ii) capability index, and (iii) vision confidence score. First, the three scores for each grasp node are collected in separate lists to find the maximum score in each category. Since the individual scores are represented in different magnitudes, they are normalized by dividing them by the corresponding maximum score in that category, so that the maximum score in each category is 1.0. This way of normalization ensures that the proportionality between the scores in each category is preserved. The three normalized scores of each grasp node are combined using geometric (multiplicative) aggregation, with all scores having equal weight. The scores that encourage the grasp success are multiplied, and the scores that hinder it are used as divisors. In this case, better capability and vision confidence are good to have, whereas a high MPV magnitude is due to clutter and is used as divisor. The geometric aggregation helps in overcoming the bias towards grasp nodes with very high or very low score in one category. Additional discussion for the chosen normalization and aggregation method can be found in [17]. After the score aggregation, the grasp nodes are sorted based on the combined score, and the best grasp node is considered for execution.

Algorithm 1 Grasp sequence planner

```
Given: World model $W$
Output: Sorted list of grasp nodes $G$
1: collect all grasp nodes from $W$ in $G$
2: init $mpv\_list$, $capab\_list$, $vision\_list$, and $grasp\_score\_list$ as []
3: for each grasp node $g_i$ in $G$ do
4:     $mpv\_list$.append($g_i['mpv\_magnitude']$)
5:     $capab\_list$.append($g_i['capability\_index']$)
6:     $vision\_list$.append($g_i['vision\_score']$)
7: end for
8: for each grasp node $g_i$ in $G$ do
9:     $mpv$ = $mpv\_list[i]$/max($mpv\_list$)
10:    $capability$ = $capab\_list[i]$/max($capab\_list$)
11:    $vision\_score$ = $vision\_list[i]$/max($vision\_list$)
12:    $grasp\_score$ = $(capability * vision\_score)/mpv$
13:    $grasp\_score\_list$.append($grasp\_score$)
14: end for
15: sort $grasp\_score\_list$ in descending order and save the indices as
    $sorted\_indices\_list$
16: reorder $G$ as per $sorted\_indices\_list$ and return $G$
```

C. Finger Stiffness Planner

In order to take advantage of the variable stiffness property of the CLASH hand, a suitable planner is required to decide the best finger stiffness for interacting with the object in a given scenario. To address this, we propose a decision tree (shown in Fig. 7) for pre-grasp interaction stiffness based on three factors:

(i) Object position uncertainty, categorized based on the confidence score provided by the vision module,
(ii) Object fragility, retrieved from the object database,
(iii) Clutter around the object, categorized based on the MPV score for the object node.

We simplify the decision process by considering only three possibilities for each factor: low, medium, or high. The resulting finger stiffness is also set to low/medium/high, which is translated into a stiffness value based on the stiffness range achievable with the hand.

The decision tree encodes key decisions for the grasp success. In case of high object position uncertainty, the stiffness is never set to high so that the fingers can try to comply as much as possible to the object lying in the vicinity of the hand. For an object with low or medium position uncertainty, the decision is based on fragility and clutter. For high clutter, it is ensured that high or medium stiffness is used depending on the object fragility, so that the fingers can pierce along the sides of the object with certain force. For a highly fragile object, it is ensured that a high stiffness is never used irrespective of the clutter. In most situations, pre-grasp compliance with the object or environment results in better grasp success. Therefore, low or medium stiffness are given more preference compared to high stiffness. A

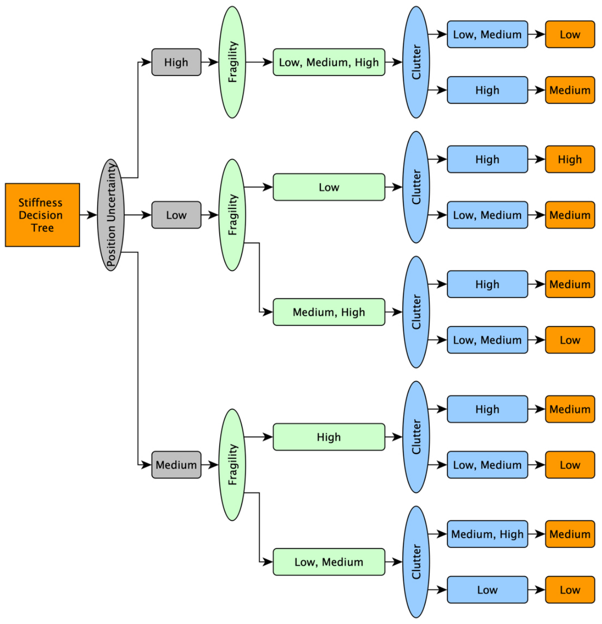


Fig. 7: Finger stiffness decision tree for pre-grasp interaction.

high stiffness is chosen only in one case with low position uncertainty, low fragility, and high clutter.

Due to the simple and direct structure of the decision tree, the planning time is negligible. As discussed in the experiments presented in Section IV-B, the choice of stiffness based on this decision tree proves to be simple, fast, and effective.

D. *Finger Position Planner*

Experiments show that grasping from a full or almost full bin is very difficult due to the constrained space for placing the fingers around the object. The stiffness decision tree helps to tackle this situation, but it is not a complete solution for grasping in high clutter. For instance, if the objects are heavy or very fragile it would not be possible to pierce along sides of the object with high stiffness. In this case, an additional local planner for finger placement can greatly improve the success rate of top grasps. The finger position planner is invoked only if 80% of the bin is filled with objects (evaluated based on depth image analysis), or if the object neighbor MPV magnitude is above 90% of the length of the hand interaction region (Fig. 3a).

The idea behind the finger position planner is to find a suitable rotation for the hand such that the fingers in the pre-grasp configuration are aligned with free regions around the object, and they can then slide in easily around the object. First, the bounding box (BB) of the object is obtained using the depth image. An expanded BB around the object is then considered, by extending the original one by three times the finger width in all dimensions. The depth image is cropped using this expanded BB, and it is preprocessed (binarized, eroded, and dilated) to remove any noise. The positions of the three fingers in the initial grasp pose are checked against the preprocessed image to verify if they fall on free spots. If there is some collision, then the mirrored positions (i.e.

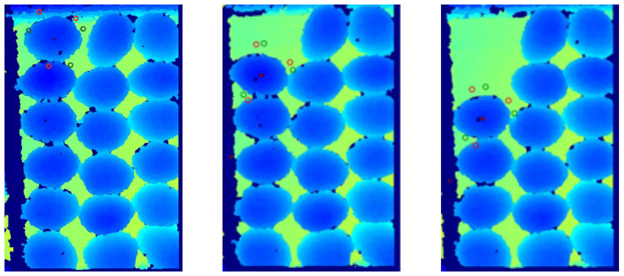


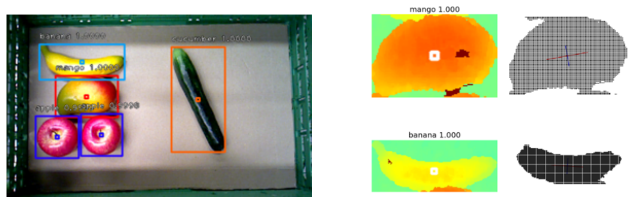
Fig. 8: Finger position planner results for the clutter experiment (Section IV-B.2). Initial and planned finger positions are shown as red and green circles.

180^o rotation of the hand) are checked. If this is also not free, then the planner starts incrementing the hand rotation in steps of 1^o until a free region is found for all three fingers. Depending on the shape primitive of the object, the allowed rotation range is different. There is no limitation for objects with spherical shape primitive, but for objects with cylindrical or cubical primitive, a maximum of 10^o rotation is allowed, to prevent the fingers from aligning with the major axis of the object, leading to a non force-closure grasp. If there is no solution for any hand rotation, the metacarpal joints of the hand are opened further (by 2^o in each iteration) to push the initial finger positions further apart. These new positions are checked for all hand rotations until a free region is found. The planner stops after three such iterations of hand openings and checking all rotations in each iteration. In case of no solution, the initially planned grasp is executed with the appropriate hand stiffness. As an example, Fig. 8 shows the output of the finger position planner for a bin completely filled with mangoes. This result corresponds to the experiment discussed in Section IV-B.2. The RGB image of the bin in this case can be found in Fig. 10a.

III. System Overview

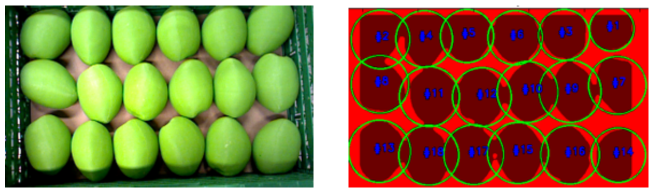
The hardware setup (Fig. 1) consists of the DLR Light-Weight Robot arm (LWR) and the CLASH hand. The robot arm controller runs on a Linux realtime machine. The other software components are executed in a separate Linux PC with Intel Xeon E5-1630 v4 CPU @ 3.70GHz. It also has a NVIDIA Quadro K620 GPU used by the vision algorithm. An Asus Xtion RGB-D sensor is used to perceive the environment. A shopping GUI provides the possibility to order items that need to be grasped from the bin. It is also used to update the user with the current state of the robot actions. The world model is created using a Neo4j graph database.

The LWR robot arm provides interfaces for both Cartesian and joint impedance control. A simple RRT-based motion planner is used for the arm. The CLASH hand also provides interfaces to control the passive finger stiffness and joint positions independently. The grasps provided by CLASH are adaptive, i.e. the metacarpal joints close until a contact with the object is achieved, and then the distal joints start closing around the object. In this way, the object is caged between the fingers, and by increasing the passive hand stiffness the objects can be grasped without applying excessive force on



(a) Bounding box detection (b) 6D pose estimation

Fig. 9: Learning-based object detection, and 6D pose estimation.



(a) RGB image (for reference) (b) Segmentation on depth image

Fig. 10: Segmentation of objects using watershed algorithm.

them. In addition, the CLASH hand provides grasp success observers based on finger torque monitoring and a proximity sensor in the palm. More details on the CLASH hand and its error recovery capabilities can be found in [14], [18].

The single shot multi-box detector [19], which uses a single deep neural network, is used for detecting object bounding boxes in the RGB image. The detections are provided with a confidence score. To estimate the 6D pose of the object, the bounding box from the RGB image is used to crop the depth image. After basic pre-processing and noise removal, the orientation of the object is computed using the eigenvector of the depth data in the cropped image. The distance of the object with respect to the camera is calculated from the central region of the bounding box in the depth image. The camera calibration matrix is used then to find the 3D position of the object. The results of bounding box detection and 6D pose estimation are shown in Fig. 9. This vision solution works well for most cases, but for highly cluttered scenes (Section IV-B.2) the performance is not satisfactory. So only for the scope of high-clutter experiments, a simple segmentation of objects in the depth data is done using watershed algorithm (Fig. 10). Here there is no classification of objects, therefore all detected segments are assumed to be of the same type (unmixed bins). The 6D pose is estimated in the same fashion as the above method.

For the overall system, the execution process is controlled with a state machine that starts after an order is received in the shopping GUI. Then, the vision algorithm is used to detect the various objects in the scene. The location of the environmental constraints (wall and edge) is always fixed (not detected automatically). The world model is then created including all the ordered objects, and the grasp sequence planner returns the best object to grasp first. The stiffness setting for the pre-grasp interaction is also computed at this stage. The finger position planner is employed in case of high clutter. After the planning stage, the preferred grasp is executed and the object is moved to the delivery bin. In case of grasp failure or if there are ordered objects still to be retrieved, the state machine reiterates starting from the object detection stage. The state machine ends when all ordered objects are retrieved and placed in the delivery bin, or if a grasp attempt was made at least three times on all ordered objects and there is no further option to retrieve them.

IV. Experimental Evaluation

This section presents a number of experiments to validate the proposed grasp strategy planning framework under different circumstances. Although the proposed planner is generic, it is discussed in the context of a grocery handling use case as an example. Six different fruits and vegetables are arranged in a structured clutter in a narrow bin. Specific items ordered by the customer need to be grasped and moved to a different bin for delivery. The attached video shows real-time execution of these experiments.

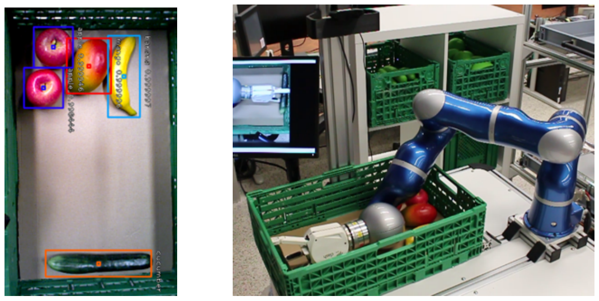
A. *Experiment I: Environmental Constraints*

This first set of experiments was designed to show that the planner is not biased and can choose all kinds of grasps, depending on the current circumstances.

1) Top Grasp: Fig. 9a shows the position of the objects in the bin for this experiment. All the objects are selected in the shopping GUI for grasping. The world model for this scenario is shown in Fig. 6. The wall grasp node is added only for the cucumber, as it is the only object having a collision-free path to the wall constraint (the position of the wall constraint is highlighted in Fig. 1). Since there is no edge constraint in this scenario, no edge grasp nodes are added. All the objects can be grasped with top grasps. The planner in this case ranked the top grasp on the cucumber as the first choice, as it is the only object with no neighbors, and also it is located in a region with acceptable arm dexterity. The wall grasp on cucumber is not the preferred choice since the EC (wall) is far from the current object position.

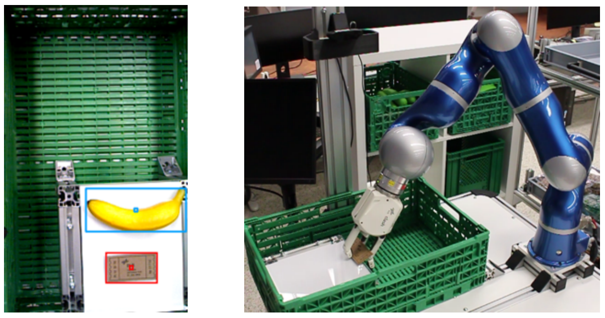
2) Wall Grasp: Fig. 11a shows the position of the objects in the bin for this experiment. The scenario is quite similar to the previous one, except that the cucumber is now very close to the wall rather than in the middle of the bin. For this scenario, the planner ranked the wall grasp on cucumber as the first choice. In fact, the top grasp for cucumber is no longer a good choice, as the wall is a very close neighbor hindering the top grasp. However, the object is very close to the EC (wall), and the bin region in the middle (required for the hand-arm interaction region) is collision-free. The robot performing a wall grasp on cucumber is shown in Fig. 11b.

3) Edge Grasp: Fig. 12a shows the position of the objects in the bin for this experiment. The goal is to grasp both the banana and the ticket. A small table is now placed inside the bin in order to have an edge constraint, which is defined only on one side of the table. The ticket can only be grasped using an edge grasp according to the semantic information recovered from the object database. In the first iteration, the world model is updated only for the banana (Fig. 13a), as the edge grasp is not feasible for the ticket because the banana is blocking the path required for this grasp. The planner returns the only choice of top grasp on banana as the preference. After the grasp on banana is successful, in



(a) RGB image (b) Robot performing a wall grasp

Fig. 11: Wall grasp experiment (Section IV-A.2).



(a) RGB image (b) Robot performing a edge grasp

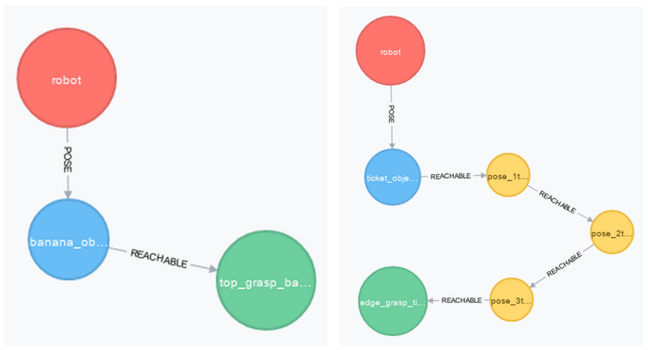
Fig. 12: Edge grasp experiment (Section IV-A.3).

the second iteration the edge is free and therefore the world model includes now the ticket object with the edge grasp node (Fig. 13b). The planner now returns the edge grasp on ticket as the preference; the robot performing this grasp is shown in Fig. 12b. This experiment shows also how the feasibility check corresponding to each object and grasp type is carried out before updating the world model.

B. *Experiment II: Variable Stiffness Hand*

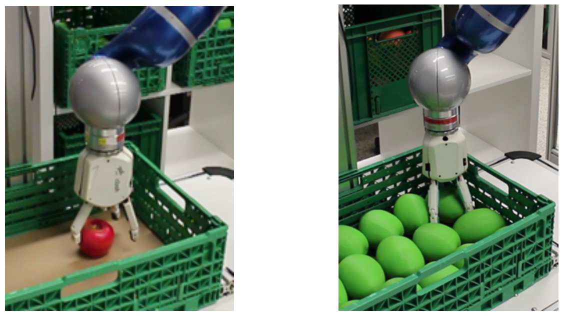
The benefits of using a variable stiffness hand and appropriate stiffness planning are demonstrated in the following experiments.

1) Object position uncertainty: Fig. 14 shows the position of the apple in the bin for this experiment. After the object pose is estimated by the vision algorithm, an error of 3 *cm* is artificially introduced in one axis (along the longest side of the bin). The confidence score of the vision algorithm is also lowered. Based on the stiffness decision tree (Fig. 7), an object with high object position uncertainty and low clutter requires a low finger stiffness. Thanks to this setting, the hand approaches the object and the fingers can adapt to its shape. Fig. 14a shows that the object is not in the middle of the hand, but still the thumb was able to accomodate due to the low stiffness, resulting in a successful grasp.



(a) Iteration I: Banana (b) Iteration II: Ticket

Fig. 13: World models created during the edge grasp experiment.



(a) Position Uncertainty (b) Clutter

Fig. 14: Use of variable stiffness for grasping (Section IV-B).

2) Clutter: A bin full of mangoes (Fig. 10a) is used for this experiment. The mangoes used in this experiment are 3D printed, but have similar dimensions and weight compared to a real mango. A watershed algorithm is used here to segment and estimate the pose of all mangoes in the bin (Section III). The finger position planner is also invoked in this case, as the bin is completely full; the results of this planner at different stages of the experiment are shown in Fig. 8. The computation time for the finger position planner in this experiment is around 600 milliseconds. Based on the stiffness decision tree (Fig. 7), for a low object position uncertainty, with medium object fragility and high clutter, the finger stiffness is set to medium. With this stiffness, the hand has the ability to pierce between the objects. The grasp sequence planner in this case decides the best object primarily based on the capability index, since all the objects have a similar set of neighbors. But after each grasp a free spot is created, and in the next iteration the sequence planner prefers mangoes with a nearby free spot. By combining the results of both the finger position planner and finger stiffness planner, the robot was able to successfully grasp multiple mangoes from the full bin (Fig. 14b).

C. *Experiment III: System Evaluation*

In this experiment, the full capability of the system is tested. The bin is filled with 12 objects randomly placed (60% of the bin is occupied), and a request to grasp five objects is made through the shopping GUI. The experiment is repeated 10 times, and the complete system had a 90% success rate. However, failures come from different elements in the overall system, not specific to the grasp strategy planner. In most cases the failure is due to the limitation of the vision algorithm to precisely classify and estimate the pose of objects in such cluttered scenes. Failures also occur due to the open loop execution of moving the objects towards the EC. The outcome of the grasp sequence planner while considering the cumulative score (capability, neighborhood, and vision), compared to considering only the capability or

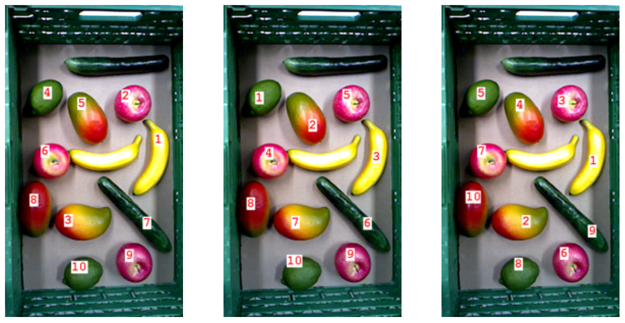

(a) Overall (b) Capability (c) Neighborhood

Fig. 15: Grasp sequence returned by the planner.

neighborhood, is shown in Fig. 15. The computation time for the creation of grasp-centric world model and grasp sequence planning in this experiment is around 500 ms, or approximately 40 ms per object in the bin.

V. Conclusion

This paper introduced a grasp strategy planner that can exploit environmental constraints for grasping objects from a bin. The planner can consider multiple ECs and can process them equally in the decision making process. The decision process for choosing the best object to grasp and the EC to use is made after considering multiple factors such as neighborhood, arm capability and vision confidence score. The decision on what object to pick up first in a cluttered scenario is especially critical, considering that there is reduced space around the objects to be picked. After each object is picked, the clutter diminishes, and different grasp strategies, such as given by the finger position planner or exploitation of an appropriate EC, can be possible. Grasping in this complex bin picking tasks was possible also due to CLASH hand and its variable stiffness property. Various experiments showed that the different components of the planner helped in successfully grasping objects in diverse scenarios. A direct comparison with other bin picking solutions is not meaningful since they do not consider the exploitation of ECs in the decision making process. Although objects like a thin ticket or a very broad book can be grasped by a suction gripper, such a gripper is not ideal for many manipulation tasks. Therefore, it is important to consider EC in the grasp decision making process in order to enable standard mutipurpose mutifingered grippers to grasp wide variety of objects in everyday scenarios.

In this work, a simple traditional grasp planner that returns a single grasp along the major axis of the object for each EC is used as an input to the grasp strategy planner. If a different grasp planner that returns multiple grasp possibilities for each EC is used, the proposed framework requires no changes. Instead of comparing one grasp per EC per object, the object sequence planner would compare multiple grasps per EC per object. Except for the planning time there would be no difference whatsoever.

This planning framework can be improved in several ways. For instance, the finger stiffness was planned with fixed rules, but other techniques such as using machine learning approaches for setting this stiffness can be explored. The current framework does not have the ability to isolate objects by using pre-grasp manipulation actions, e.g. pushing other objects, in order to make the object of interest graspable. Since the framework already provides information on neighbors and which objects hinder the path to a desired grasp, extending it with pre-grasp actions is a next step. Grasping from a clutter pile is also a natural extension to this work. To achieve this, the MPV computation for neighboring objects and the collision free path to move the object to the EC must be tackled in 3D.

References

[1] A. Sahbani, S. El-Khoury, and P. Bidaud, “An overview of 3D object grasp synthesis algorithms,” *Robotics and Autonomous Systems*, vol. 60, no. 3, pp. 326–336, 2012.

[2] M. A. Roa and R. Suárez, “Grasp quality measures: review and performance,” *Autonomous robots*, vol. 38, no. 1, pp. 65–88, 2015.

[3] M. Laskey, J. Lee, C. Chuck, D. Gealy, W. Hsieh, F. Pokorny, A. Dragan, and K. Goldberg, “Robot grasping in clutter: Using a hierarchy of supervisors for learning from demonstrations,” in *IEEE Int. Conf. on Automation Science and Engineering - CASE*, 2016, pp. 827–834.

[4] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen, “Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection,” *Int. J. Robotics Research*, vol. 37, no. 4-5, pp. 421–436, 2018.

[5] L. Pinto and A. Gupta, “Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours,” in *IEEE Int. Conf. on Robotics and Automation - ICRA*, 2016, pp. 3406–3413.

[6] M. Dogar and S. Srinivasa, “A framework for push-grasping in clutter,” in *Robotics: Science and Systems*, 2011.

[7] M. Dogar, K. Hsiao, M. Ciocarlie, and S. Srinivasa, “Physics-based grasp planning through clutter,” in *Robotics: Science and Systems*, 2012.

[8] B. Sauvet, F. Lévesque, S. Park, P. Cardou, and C. Gosselin, “Model-based grasping of unknown objects from a random pile,” *Robotics*, vol. 8, no. 3, 2019.

[9] D. Berenson and S. S. Srinivasa, “Grasp synthesis in cluttered environments for dexterous hands,” in *IEEE-RAS Int. Conf. on Humanoid Robots*, 2008, pp. 189–196.

[10] F. Heinemann, S. Puhlmann, C. Eppner, J. Álvarez-Ruiz, M. Maertens, and O. Brock, “A taxonomy of human grasping behavior suitable for transfer to robotic hands,” in *IEEE Int. Conf. on Robotics and Automation - ICRA*, 2015, pp. 4286–4291.

[11] C. Eppner and O. Brock, “Planning grasp strategies that exploit environmental constraints,” in *IEEE Int. Conf. on Robotics and Automation - ICRA*, 2015, pp. 4947–4952.

[12] C. Piazza, G. Grioli, M. Catalano, and A. Bicchi, “A century of robotic hands,” *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 2, pp. 1–32, 2019.

[13] S. Wolf, G. Grioli, O. Eiberger, *et al.*, “Variable stiffness actuators: Review on design and components,” *IEEE/ASME Trans. on Mechatronics*, vol. 21, no. 5, pp. 2418–2430, 2015.

[14] W. Friedl, H. Höppner, F. Schmidt, M. Roa, and M. Grebenstein, “CLASH: Compliant low cost antagonistic servo hands,” in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems - IROS*, 2018, pp. 6469–6476.

[15] O. Porges, T. Stouraitis, C. Borst, and M. A. Roa, “Reachability and capability analysis for manipulation tasks,” in *ROBOT2013: First Iberian Robotics Conference*, 2014, pp. 703–718.

[16] D. Leidner, C. Borst, and G. Hirzinger, “Things are made for what they are: Solving manipulation tasks by using functional object classes,” in *IEEE-RAS Int. Conf. on Humanoid Robots*, 2012, pp. 429–435.

[17] C. Tofallis, “Add or multiply? a tutorial on ranking and choosing with multiple criteria,” *INFORMS Trans. on Education*, vol. 14, no. 3, pp. 109–119, 2014.

[18] W. Friedl and M. A. Roa, “CLASH -a compliant sensorized hand for handling delicate objects,” *Frontiers in Robotics and AI*, vol. 6, 2019.

[19] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, “SSD: Single shot multibox detector,” in *Europ. Conf. on Computer Vision - ECCV*, 2016, pp. 21–37.