2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Enhancing Object Grasping Efficiency with Deep Learning and Post-processing for Multi-finger Robotic Hands

Pouya Samandi[1], Kamal Gupta[2] and Mehran Mehrandezh[3]

Abstract— **This paper builds upon the well-established ML-based grasping technique, known as the Grasp-Rectangle (GR) method. The original GR method made two simplifying assumptions: it was designed exclusively for two-finger grippers, and it assumed that the gripper would approach objects solely from a top-down perspective on a horizontal surface. We have extended the GR method, for a multi-finger hand beyond these assumptions to (1) enable grasping from top and side views and (2) engage multiple points of contact, enhancing the algorithm's overall performance. Our approach leverages geometric cues extracted from object images to calculate the optimal grasp pose and contact points, thereby enhancing grasp reliability. Extensive testing was conducted using a 7-DOF robotic arm equipped with a 7-DOF 3-finger gripper. We achieved an accuracy of 98.6% on the Cornell Grasping Dataset with a processing time of 120 milliseconds. Furthermore, when assessing object grasping from both top and side perspectives, our algorithm delivered successful grasps at rates of 95% and 96%, respectively. These findings are rooted in a comprehensive series of tests performed across a diverse array of objects.**

I. INTRODUCTION

A captivating challenge in the world of robotics is to develop effective grasping techniques. Over time, researchers explored different approaches, such as analytical methods [1] [2] [3], geometric shape-based grasping [4] [5] [6], and machine learning-based methods [7] [8] [9]. However, many of these methods were limited to specific tasks or objects, lacking the flexibility required for handling diverse scenarios [1] [4] [8]. In recent years, the focus of research has shifted towards creating grasping systems with greater adaptability, enabling robots to handle a wide range of objects, regardless of their shapes and sizes [7] [10] [11]. To achieve this, machine learning algorithms have emerged as invaluable tools for improving grasp accuracy and reducing processing time [12] [13] [14]. While machine learning algorithms have shown impressive results in controlled experiments, they may not always be entirely reliable, as their performance heavily relies on the quality of the dataset and training process [15].

In this paper, we build upon the well-established ML-based grasping technique, known as the Grasp-Rectangle (GR) method. The original GR method made two simplifying assumptions: it was designed exclusively for two-finger grippers, and it assumed that the gripper would approach objects solely from a top-down perspective on a horizontal surface. We have extended the GR method, for a multi-finger hand beyond these assumptions to (1) enable grasping from various angles and (2) engage multiple points of contact, enhancing the algorithm's overall performance.

In summary, our work introduces several contributions to the field of robotic grasping: a) it explores the uncharted territory of applying the Grasp Rectangle method to multi-finger robots, b) this approach doesn't rely on fixed datasets, and it is generalizable for a wide variety of objects. Additionally, c) we go beyond the traditional top-down grasping of the grasp rectangle method, and we explored grasping in three dimensions for more flexible grasp choices. Moreover, d) our method improves the Grasp Rectangle Network and uses a smart search algorithm to ensure accurate and precise grasp planning. Finally, e) its easy implementation and simplicity make it accessible, enabling quick integration into real-world applications and encouraging practical use with any multi-fingered hand.

II. RELATED WORKS

Researchers have explored various methods for improving multi-finger grasping [7] [12] [13] [16]. Some have used object and hand geometry [3], while others rely on feature extraction techniques to analyze local shape descriptors and match them with hand templates [4]. Additionally, some have combined geometric approaches with Convolutional Neural Networks (CNNs) [12] [13] [17]. For instance, [7] developed a CNN that predicts stable grasp positions using partial point clouds of objects, while [8] introduced an active deep learning approach to address data collection challenges for multi-fingered hands. While these techniques offer valuable insights, they can be time-consuming [8] [12] [13] and may not always meet industry standards in terms of reliability. (see Table II).

On the other hand, researchers have developed faster and more reliable methods for grippers and two-finger robots to grasp objects [11] [16] [18]. One of these techniques is the grasp rectangle method [10] [19]. This technique creates a rectangle that defines five key parameters: the center of grasp (x and y coordinates), orientation (θ), and the width of finger opening (w), to grasp an object as shown in Fig. 1. These parameters define the position, alignment, and gripping configuration required for effective grasping [10]. Recent research leveraged (CNN) models to enhance both accuracy and processing speed [16] [18] [20] [21].

A key paper that uses the grasp rectangle method is [11]. In this paper, a modular robotic system is presented

[1]Pouya Samandi is with the School of Engineering Science, Simon Fraser University, Burnaby, BC V5A 1S6, Canada samandi.pouya@gmail.com

[2]Kamal Gupta is with School of Engineering Science, Simon Fraser University, Burnaby, BC V5A 1S6, Canada kamal_gupta@sfu.ca

[3]Mehran Mehrandezh is with Faculty of Engineering and Applied Science, University of Regina, Regina, SK S4S 0A2, Canada Mehran.Mehrandezh@uregina.ca

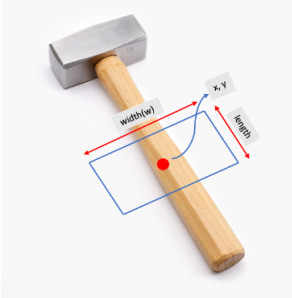


Fig. 1: Grasp Rectangle parameters

to address the challenge of generating and executing effective grasps on unknown objects. The approach involves a novel model called Generative Residual Convolutional Neural Network (GR-ConvNet) that can produce antipodal grasps from RGB-D input in real-time (around 20 milliseconds). However, there are problems with using this network for robots with multiple fingers, as explained in Section III-A. Additionally, because the dataset this network trained on (Cornell [22] and Jacquard [23]) consists of images that have been taken from above only (top), this network faces challenges when dealing with objects in complex backgrounds or views other than top. To tackle these challenges with greater efficacy, we've introduced an innovative post-processing technique integrated into the GR-ConvNet network. Our technique operates on segmented images and extends the grasp rectangle method, specifically tailored for multi-fingered robots. In addition, our method incorporates a robust search-based approach, enhancing the network's reliability and adaptability. We call our search based method, SB_ConvNet.

III. Problem Description

Our project aims to develop a robust algorithm for a multi-finger robot to successfully grasp various objects. We specifically focus on objects within the robot's workspace that are visible to an eye-in-hand camera, i.e., Fig. 12 shows a representative image of the object seen by this eye-in-had camera. The robot utilized in our project is a Kinova Gen 3 robotic arm (7 DOF) [24], paired with a Schunk SDH three-finger robotic hand (7 DOF) [25] responsible for the grasping task. Each finger on this hand features 2 DOF, with a pivoting joint situated between fingers 2 and 3, facilitating finger pivoting (Fig.7). This effort is particularly concentrated on objects observed either from a top perspective (shown in Fig. 12.a) suitable for bin-picking, or from the frontal angle (named as from side grasping shown in Fig.12.b). Additionally, we're focusing on grasping a single object at a time, i.e., for each trial, there is only one object in the camera view to grasp.

Also, we are using tactile sensing feedback, available in SDH fingers, to regulate the applied pressure for stable grasps. The primary objective of the feedback mechanism is to ensure a gradual reduction in finger closing speed as the pressure on the object increases. By slowing down the finger movement, we can maintain a delicate balance between exerting enough force to secure the object and preventing excessive pressure that could potentially lead to damage. This allows for precise control over the applied force, ensuring a safe and controlled grip on the object.

To grasp objects effectively, we need to consider five parameters: $\{P_r, \theta_r, w, \theta_p, G_t\}$. P_r [x, y, z] represents the arm's end-effector positions, while θ_r [θ_x, θ_y, θ_z] denotes its orientation. These parameters are used to solve inverse kinematics and determine its joint positions. For simplification, we assume that the gripper's palm is perpendicular to the object's face, keeping θ_x (yaw) and θ_y (pitch) constant. This allows us to define the end-effector's Cartesian pose as $\{P_r, \theta_z\}$, effectively a 4-DOF pose. The 'w' parameter defines the optimal initial finger separation width (based on object size) before grasping an object and the θ_p denotes the pivoting joint angle. Lastly, 'G_t,' the grasp type parameter, specifies the desired grasp method: lateral, tripod, or power grasp (as shown in Fig. 2).

We use success rate as the key metric to test the grasps. A successful grasp is defined as lifting the object from the initial grasp position and maintaining it at the lifted position for 1 second. The success rate is the percentage of successful grasps to the total attempted grasps. Furthermore, the term accuracy of a grasp is defined based on the Intersection over the Union (IoU) method explained in section V.A.

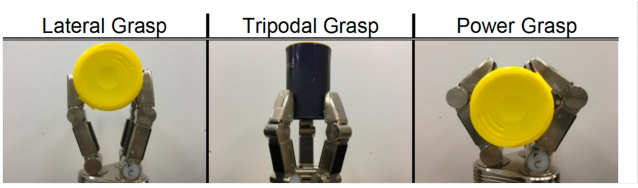


Fig. 2: Different Grasp types

A. Grasping objects without any post-processing

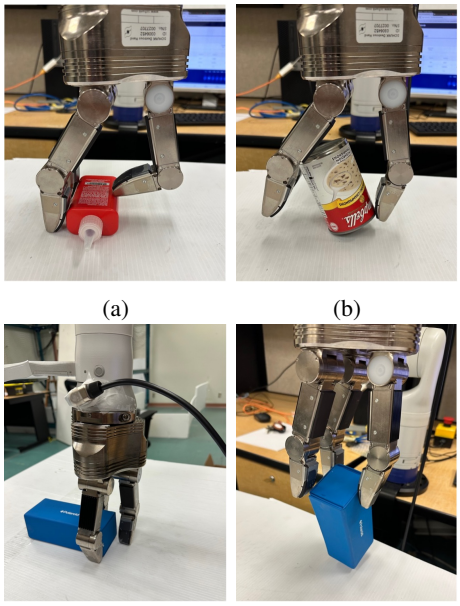
We try to grasp objects solely relying on the rectangle provided by the GR-Conv [11]. We focus on extracting essential parameters : $\{P_r, \theta_z, w, \theta_p\}$ while omitting the detailed procedures given in [11]. We set θ_p=0 and G_t=tripod due to limited available information. Our initial assessment yielded a modest 65% success rate testing on 10 different objects and 10 attempts for each object. We identified three primary reasons for these outcomes as follows:

1) Unbalanced Pressure: Fig. 3.a demonstrates a scenario where uneven pressure was applied to the object, potentially causing its displacement from the intended grasp location or, in more severe instances, leading to the object falling (Fig. 3.b). This highlights the importance of calibrating the grasp center in the horizontal direction (x) value to put even pressure on the object at contact points.

2) Missed Contact: Fig. 3.c illustrates a situation where the grasp placement was too close to the object's edges or even outside its boundaries. This often resulted in a finger failing to make contact with the object, resulting in missed grasps. This highlights the importance of calibrating the grasp center in the vertical direction (y) value for all the finger contact assurance.

3) Slippage due to rotational forces: As depicted in Fig. 3.d, instances arose where the object slipped from the robot's grip due to the rotational force-couple exerted by the object's center of mass. This highlights the significance of aiming to grasp objects closer to their center of mass.

As a result, to solve the proposed challenges, we offer a post-processing algorithm to ensure the accuracy of the grasp for a multi-finger robot.



(a) (b) (c) (d)

Fig. 3: a) unbalanced pressure had pushed the object to another position, b) Unbalanced pressure resulted in the object's fall down, c) one finger missed the object while it was closing, d) slippage of the object from the hand

B. Hand Closure Methodology

In this project, we define three distinct hand poses for grasping objects: Lateral, Tripod, and Power, as illustrated in Fig. 2. To execute these grasps effectively, we rely on three key parameters: pivotal joint angle (Fig. 7), hand initial width opening (parameter "w" defined by the GR), and grasp type. Manipulating these parameters allows precise control of the robotic hand's movements for grasping various objects. Once the fingers are in their initial positions, we experimentally adjust the joints' speed to achieve the final grasp-type configuration of the fingers.

IV. Grasp Strategy

As previously mentioned, we modify the GR-Convnet for precise grasp point determination. To enhance its effectiveness, we incorporate segmented imagery as input, a task-efficiently accomplished using the GrabCut technique from OpenCV. GrabCut offers a simple implementation process while maintaining satisfactory and reliable accuracy for our specific objectives. The GrabCut method necessitates an initial bounding box drawn around the target object for an approximate localization within the image. To create this bounding box, one effective approach involves employing object detectors like YOLO. In our case, we fine-tuned YOLOv5 [26] using a household objects dataset [27], narrowing down its focus to 20 common household items. Impressively, after fewer than 50 training rounds, our model achieved an accuracy of 98.24% on the evaluation dataset and 97% on the test dataset. For more in-depth details, we refer readers to [28] [29]. In the real-world, our object detector demonstrated a 98.8% success rate in detecting objects for side grasps. However, relying solely on YOLO to create the bounding box has its limitations, particularly for grasping from the top. This decline is attributed to the limited variety of objects in the household dataset.

Conversely, the GRConvNet excelled in identifying grasp rectangles for a broader range of objects, accommodating various shapes and sizes. Nonetheless, this method exhibited vulnerability to changes in background color and depth images. Instances with vivid backgrounds or abrupt depth changes could impact the accuracy of this approach in generating feasible grasp rectangles. In top-view grasping scenarios, conditions typically involve a consistent, uniform background with stable depth maps as objects are placed on a level surface. However, side-view grasping situations present challenges with depth maps that exhibit abrupt variations and diverse, vibrant backgrounds. This discrepancy can be traced back to the training datasets, such as the Cornell [22] and Jacquard [23] datasets, which primarily featured objects on monochromatic surfaces at specific camera-to-object distances.

As a result, our approach leverages the YOLO detector's proficiency in creating bounding boxes for the side-view grasping dataset, while the GR network adeptly generates these bounding boxes for the top-view grasping context (Fig. 5. Also, See the accompanying Fig. 4 for an illustration of the roadmap to determine all the necessary parameters for successful object grasping.

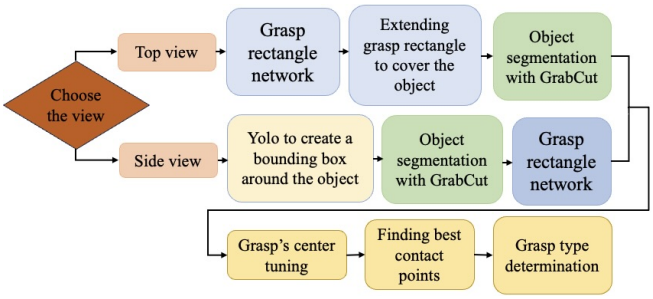
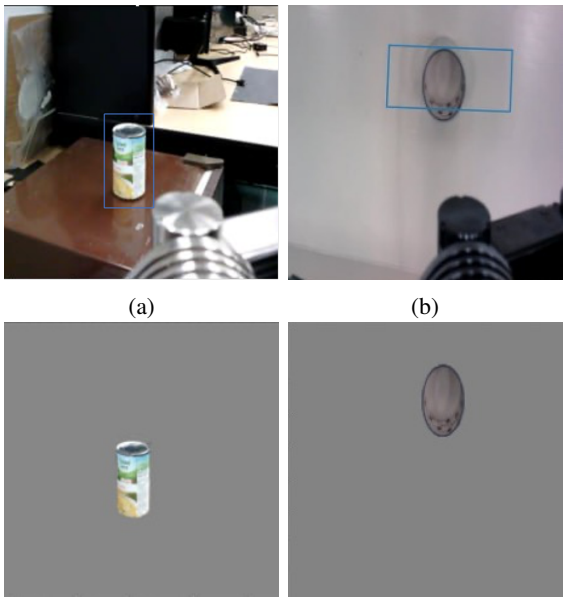


Fig. 4: Process of Grasping objects in different views

A. Grasp Center(x,y) tuning

As discussed in section III-A, to achieve better results, we need to fine-tune the grasp center parameters in the image. To elaborate on this technique, we offer Fig. 6 as a visual guide. First, we rotate the image by θ degrees around the center of the image to ensure that the object is positioned from top to bottom.θ is extracted from the grasp rectangle network and represents the rotation of the rectangle with respect to the horizontal coordinate of the image. As shown in Fig. 6.a, we transfer the x_c to the point in the middle of the right and



(a) (b)

(c) (d)

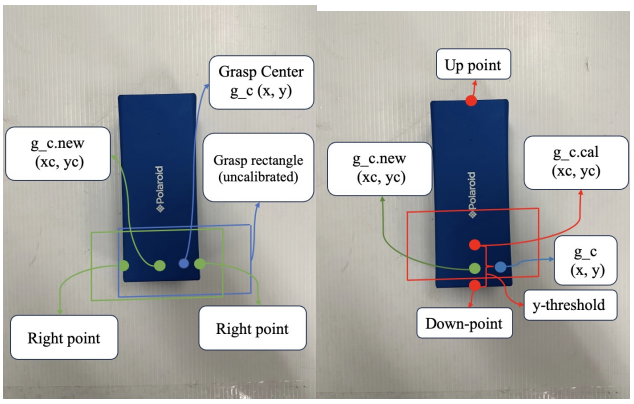
Fig. 5: a) bounding box created by YOLO (side view), b) bounding box created by grasp rectangle method (top view), c) segmented object (side view) and d) segmented object (top view)

left points and we name it as g_c.new. These points sit at the object's extreme edges and share the same vertical position (y) as the "grasp center (g_c)".

Next, we focus on calibrating the y-coordinate. Our goal is to ensure there is enough space between the g_c.new and the vertical edge of the object. To achieve this, we measure the distance between the g_c.new and the "Up Point" and "Down Point". These points sit at the object's extreme edges and share the same horizontal position (x) as g_c.new. If these two distances happen to be smaller than a predefined threshold, we transfer the g_c.new in a direction that meets the threshold and we call the calibrated point "g_c.cal". In addition, when the object is small, and the distance between the grasp center point and both the 'up point' and 'down point' falls below the threshold, we position the grasp center at the midpoint of these extreme points. This step guarantees that all the fingers have adequate space to potentially make contact with the object. Nevertheless, this calibration process aims to optimize our grasp without significantly shifting the grasp rectangle from its initial position. This way, we can maintain the highest grasp quality based on the output of the original Network.

B. Finding the Best Contact points

We use a search-based algorithm to find the best way to grasp an object. One should note that our method can be applied to any multi-finger robot. This involves evaluating different grasp candidates and selecting the most effective one. To start, we need to estimate where the fingers would make contact when we place the palm over the center of the grasp and close the fingers. This requires determining the position of the fingertips within the image, as depicted in

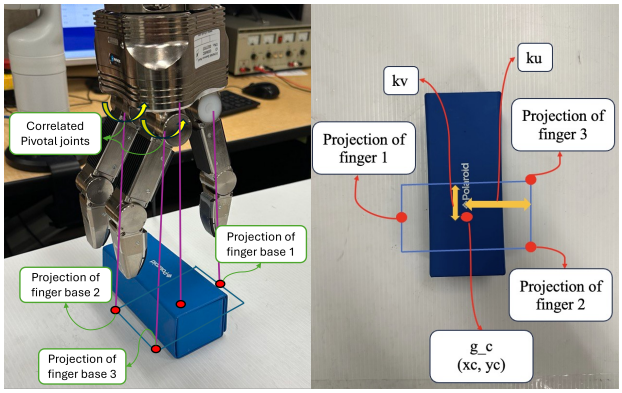


(a) (b)

Fig. 6: Process of calibrating grasp center

Fig.7.a. In Fig.7.b, the projection of fingers creates mirrored points of the fingertips on the image plane. k_u and k_v are experimental values representing the rectangle parameters used to project the fingertips onto the ground. We experimentally calculate the k_u and k_v values based on a linear equation (Equitation (1)). The Z_c is the distance between the camera frame and the object in mm and Z_0 and k_0 are constant parameters that need to be calibrated for each robot and camera system. For our system, these values were about 600 mm and 20 mm, respectively.

$$k_v = k_u = k_0 - (Z_c/Z_0 - 1) \tag{1}$$



(a) (b)

Fig. 7: Projecting hand's fingertips on the grasp plane.

Analyzing fingertips enables the establishment of a line indicating finger closing direction, intersecting with the object's edge to reveal contact points for each finger. Recall that the pivotal joint (θ_p) allows rotation of Fingers 2 and 3 (around an axis perpendicular to the palm), and it will likely alter the contact points.

Fig. 8 serves as an example to help better understand the concept. In the Figure, $\hat{x}$ represents the coordinate along the horizontal axis of image coordination, while $\hat{y}$ denotes the coordinate along the vertical axis. We let the pivoting joint angle rotate in a range from 0 to 90 degrees in 10-degree intervals. Hence index j takes values in a range from 0 to 9. Thus $\theta_{p,j}$ denotes finger p ($p = 2$ or 3), at

pivotal angles corresponding to j. For example, $\theta_{2,0} = 0^\circ$ and $\theta_{2,4} = 40^\circ$ for finger 2. This applies to finger 3 as well. The base frame of finger 1 remains fixed, hence $j = 0$ always, i.e., $\theta_{1,0} = 0^\circ$ and it can only be closed in one direction, which is the horizontal direction. Now, let $f_{p,j}$ indicate the direction of finger where $p \in \{1,2,3\}$ indicates the finger's number and j^{th} closing direction of the pivotal joint where: $j \in \{0,1,...,9\}$ for fingers 2 and 3, $j = \{0\}$ for finger 1. For finger 2, $f_{2,j}$ equals $\theta_{p,j}$, and for finger 3, $f_{3,j}$ equals $-\theta_{p,j}$. $CP_{p,j}$ refers to the contact point where the closing finger p makes contact with the object, exerting sufficient pressure to halt its movement. $n_{p,j}$ denotes the normal direction of the surface at the contact point for finger p at the j^{th} pivoting joint angle. Our objective is to minimize rotational disparities between finger directions and surface normals, quantified by $\Delta\theta_j$ (Equation 2). The $\hat{\Theta}$ (Equation 3) corresponds to the index of minimum value of $\Delta\theta$ and indicate the optimal $\theta_{p,j}$.

$$\Delta\theta_j = \sum_{p=1}^{3} |f_{p,j} - n_{p,j}| \tag{2}$$

$$\hat{\Theta} = \text{argmin } [\Delta\theta_0, \Delta\theta_1, ..., \Delta\theta_9] \tag{3}$$

For Finger 1, movement is restricted to a single direction parallel to the horizontal line ($f_{1,j} = 0^\circ$). Fingers 2 and 3 have more flexibility, pivoting relative to each other. In the Figure, the pivotal joint angles of the green and yellow arrows are 0° and 40°, respectively. It clearly illustrates that a 0° pivotal joint results in smaller $\Delta\theta$ for all fingers, aligning better with the object's surface normal. Conversely, a 40° pivotal joint yields larger $\Delta\theta$, less suitable for alignment. In essence, this method enables us to optimize finger rotation for improved alignment with the object's surface normal, determining the most effective finger configuration for a successful grasp.

In summary, our objective in this section is to determine a pivotal joint angle θ_p that minimizes the angular difference between the direction of the fingers and the normals of the contact points.

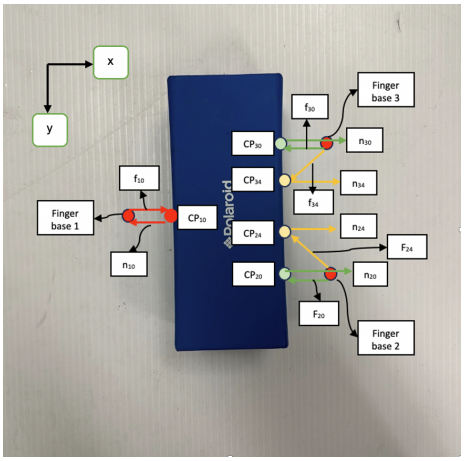


Fig. 8: Varied finger directions with different pivotal joint angles to find the optimal value.

C. Grasp Type Determination Based on Grasp Width and Height of the Object

To complete the parameter setup, determining the grasp type is paramount. In from-side grasping, we stick with the stable, power grasp. But in from-top grasping, we decide the grasp type based on the object's width and height. The grasp width indicates how wide the fingers should be opened from each other before executing the grasp. The object's height represents the height of the object at the grasp rectangle center (g_c). We create the grasp rectangle by connecting the contact points $CP_{p,\hat{\Theta}}$ for the best grasp candidate. As we adjust the rectangle's width, we define a new grasp width (Fig. 9). For height, we rely on the depth image, comparing the object's highest point to the ground. Eventually, we choose grasp type based on Table I.

TABLE I: Choosing Grasp type based on object's height and width. Please see text for explanation.

Object	Rectangle Width	Object's-Height_z	Grasp Type
Object1	Width<Lateral threshold	-	Lateral
Object2	Width>Lateral-threshold	Height<power-threshold	Tripod
Object3	Width>Lateral-threshold	Height>power-threshold	Power

Object Grasping Guidelines: In grasp planning, the choice of grasp type for an object is determined by its width and height relative to predefined lateral and power thresholds. These Thresholds are pixel sizes with values 50 and 100 respectively for our example. If an object's width is below the lateral threshold and irrespective of its height, the lateral grasp with two closely positioned fingers is recommended. When the width exceeds the lateral threshold, but the height remains below the power threshold, the tripod grasp, utilizing three fingers for stability, is deemed optimal. For objects surpassing both thresholds, the power grasp would be the most suitable, utilizing the entire hand for enhanced stability. These threshold values are predetermined based on task-specific requirements and often derived from experimental data to optimize grasp planning for diverse objects. At last, Fig.9, compares the grasp rectangle provided by GR-ConvNet, before and after post-processing.

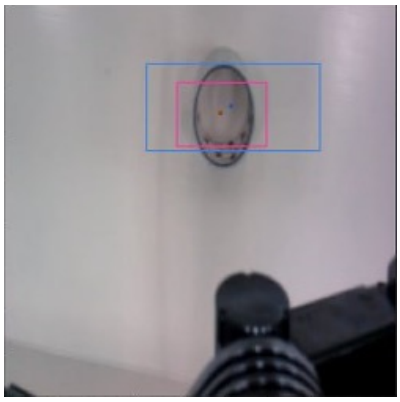


Fig. 9: Blue and Pink rectangles: pre- and post-processing grasps, respectively

V. RESULTS

For a comprehensive comparison of our searched-based method (SB_ConvNet) and the establishment of a robust benchmark, we tested our method against two distinct categories: 1) parallel gripper methods and 2) multi-finger grasping methods. To compare with a baseline of parallel gripper methods [11] [16] [20], we executed two distinct experiments. First, we compared the accuracy of rectangles generated by our SB_ConvNet method on the Cornell dataset [22] with baseline methods. Second, we conducted real-world experiments to gauge the success rate of object grasping from a top-down perspective. To ensure experiment validity, we utilized objects similar to those employed in previous studies [11] [16] [20] as some of these objects have been shown in Fig.10.

To compare with multi-finger grasping methods, we conducted additional experiments employing household objects. We assessed the success rate of grasping objects from both top and side views. Again, to ensure a fair comparison with benchmarks, we meticulously selected objects resembling those used in multi-finger methods studies [7] [9] [13] (see Fig. 11).

A. Evaluation against baseline parallel gripper methods

1) ***SB_ConvNet performance on Cornell dataset****:* We tested the output of our network and its calibration process on the Cornell dataset to compare its accuracy with some baseline approaches. To ensure a fair comparison, we adopted the evaluation technique used by other grasp rectangle methods [11] [16]. This metric defines the criteria for a grasp to be considered valid as follows: 1. The Intersection over Union (IoU) score between the predicted grasp rectangle and the ground truth grasp rectangle must be greater than 25%. 2. And that the difference in grasp orientation between the predicted grasp rectangle and the ground truth rectangle should be less than 30 degrees.

We used 5-fold cross-validation to evaluate our SB_Convnet performance. The results are reported in Table II along with the reported accuracy of some of the most accurate methods (taken from Cornell's official website [22]).

Our search-based method, SB_ConvNet emerges as a standout performer, achieving an impressive accuracy of 98.8%. This substantiates the efficacy of our calibration technique in enhancing the network's grasp detection capabilities showcases the advancement offered by SB_ConvNet and underscores its potential to enhance the precision and reliability of grasp detection in real-world scenarios. The processing time is the required computation time for extracting grasp parameters from an RGB-D image. Indeed, our model is slightly slower compared to existing methods, but still suitable for near real-time applications.

2) ***Real-word Grasping Experiments on Cornell objects****:* Although the grasping mechanisms differ between two-finger and multi-finger grippers, we demonstrate the effectiveness of our method by testing it with objects similar to those in the Cornell Dataset [22] and grasped by other grasp rectangle models [21] [20] (see Fig.10). We specifically focused on grasping objects solely from the top, consistent with prior research. The only limitation is that the objects must be at least 10 mm in height to be grasped by our relatively large hand. We conducted tests with 15 different objects, each grasped 10 times independently. As presented in Table II, column 4, our experimental results for real grasp execution demonstrated superior performance, achieving a remarkable 98% success rate. Notably, our success is partly attributed to the multiple contacts between the objects and the fingers.

Fig. 10: A sample of objects that has been selected from the Cornell dataset [22]

TABLE II: Comparison of Grasp Rectangle Methods on Cornell Grasp Dataset and Real Experiments with Comparable Objects.

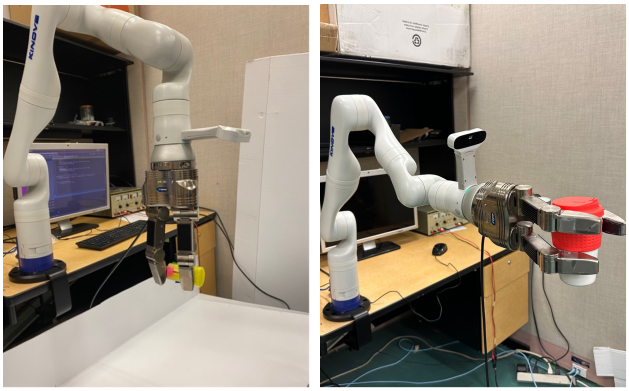
Network Name	Processing Time(ms)	Cornell accuracy(%)	Expt success rate(%)
Chu et al. [16]	14	96.0	89.0
Cheng et al. [20]	14	96.5	90.4
Wu et al. [21]	26	98.9	91.3
Kumra et al. [11]	20	97.7	95.4
Ainetter et al. [18]	20	98.2	-
SB_ConvNet (ours)	120	98.8	98.0

B. Evaluation against baseline multi-finger grasp methods

Our method outperforms parallel grippers in object grasping, however, comparison with multi-finger grasping techniques [7] [9] [13] [17] is our main concern. We conducted tests on 15 household objects, with 150 attempts for grasping from the top and an additional 15 objects with 150 attempts for grasping from the side, similar to household objects grasped in [9] [13] [17] (see Fig. 11). Some objects overlapped with those from the Cornell dataset. An example for each grasp is shown in Fig.12.

Table III compares our approach to some of the leading approaches in robotic multi-finger grasping methods - some of these are geometrically based and some use CNN models. Our SB_ConvNet significantly advances the state-of-the-art in both processing time and grasp success. Our model serves as a general grasp method, designed to handle objects of various shapes as shown in Fig.10. Also, as explained in section III, we have executed base ConvNet (Kumara et al [11]) without any post-processing and resulted in only a 65%

Fig. 11: A sample of household objects for multi-finger grasping experimental evaluation [7] [9]



(a) (b)

Fig. 12: Object grasped from a) top and b) side

success rate in grasping objects. That was the main reason that inspired us to implement our post-processing method and we achieved a 96% success rate in grasping from the top and 95% from the side. This clearly shows that our method SB_ConvNet presents a fast and reliable solution.

It's noteworthy that that the published results of methods have employed diverse robots and computing hardware systems, nevertheless the comparison with our results do show the efficacy and advantages of our approach. Our approach operates on a single image without the need for total or partial representation of object point clouds from multiple viewpoints, demonstrating speed and potential for real-time applications. Despite variations in systems, our model does achieve a significantly higher success rate and offers adaptability to other robotic hands with minor adjustments.

However, it is important to note that our method encountered challenges when attempting to grasp objects of solid white or transparent color due to limitations in the GrabCut method. Despite this limitation, our approach showed significant advancements in grasping objects of varying shapes and colors. In addition, our network showed slightly weaker results on household objects compared to objects similar to the ones presented in the Cornell grasp dataset. One reason could be that the objects used in our tests were generally bigger/heavier, and many of them were slippery and transparent. (Fig.11).

TABLE III: Comparison of grasping objects using Multi-finger robots

Network Name	Input	Processing Time(s)	Fingers Number	Success Rate(%)
Lundell et al. (2021a) [9]	Point-cloud	8.1	3	71.0
Lundell et al. (2021b) [13]	Point-cloud	9.1	3	60.0
Lu et al. [7]	RGB-D	10.2	4	75.0
Baron et al. [17]	Point-cloud	-	3	83.3
Kumra et al. [11] (ours)	RGB-D	0.20	3	65.0
SB_ConvNet (ours)	RGB-D (side view)	0.120	3	96.0
SB_ConvNet (ours)	RGB-D (Top view)	0.120	3	95.0

VI. Conclusion

In our study, we've introduced an efficient post-processing technique that harnesses a search-based algorithm to enhance the performance of the well-established "grasp rectangle (GR)" method for multi-finger hands. We have tuned our model for a 3-finger robotic hand (SDH), but our methodology can be extended to any parallel or multi-finger robot, at least at the concept level. What sets our approach apart is its remarkable speed and reliability compared to existing multi-finger robot grasping methods. Notably, our proposed post-processing technique not only elevates the outcomes of GR Networks but also, improved the reliability of grasping even for two-finger grippers.

The GrabCut method we used for segmentation, excelled in terms of speed and reliability. However, it did face challenges when detecting transparent and white objects. In our future work, we intend to implement more advanced segmentation methods to tackle these challenges and further enhance the efficiency and effectiveness of our approach. Additionally, we aim to expand our method to grasp an object amidst a cluster of other objects. Finally, in this study, for simplification, we assumed that the yaw and pitch of the end-effector pose remained constant and equal to zero. This assumption limited the generality of the grasping ability. In future work, we plan to explore a broader range of end-effector poses to enhance object grasping.

References

[1] S. El-Khoury and A. Sahbani, "On computing robust n-finger force-closure grasps of 3d objects," in *2009 IEEE International Conference on Robotics and Automation*, May 2009, pp. 2480–2486.

[2] R. Krug, D. Dimitrov, K. Charusta, and B. Iliev, "On the efficient computation of independent contact regions for force closure grasps," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, October 2010, pp. 586–591.

[3] M. A. Roa and R. Suárez, "Grasp quality measures: review and performance," *Autonomous Robots*, vol. 38, no. 1, pp. 65–88, January 2015.

[4] M. Hegedus, K. Gupta, and M. Mehrandezh, "Efficiently finding poses for multiple grasp types with partial point clouds by uncoupling grasp shape and scale." *Autonomous Robots*, vol. 46, pp. 749–767, 2022.

[5] R. R. Devaraja, R. Maskeliūnas, and R. Damaševičius, "Design and evaluation of anthropomorphic robotic hand for object grasping and shape recognition," *Computers*, vol. 10, pp. 1–14, 2021.

[6] M. Zhu, K. G. Derpanis, Y. Yang, S. Brahmbhatt, M. Zhang, C. Phillips, M. Lecce, and K. Daniilidis, "Single image 3d object detection and pose estimation for grasping," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2014, pp. 3936–3943.

[7] Q. Lu, M. V. der Merwe, and T. Hermans, "Multi-fingered active grasp learning," June 2020, accessed: August 26, 2024. [Online]. Available: http://arxiv.org/abs/2006.05264

[8] J. Varley, J. Weisz, J. Weiss, and P. Allen, "Generating multi-fingered robotic grasps via deep learning," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, September 2015, pp. 4415–4420.

[9] J. Lundell, F. Verdoja, and V. Kyrki, "Ddgc: Generative deep dexterous grasping in clutter," *IEEE Robotics and Automation Letters*, vol. 6, pp. 6899–6906, October 2021.

[10] I. Lenz, H. Lee, and A. Saxena, "Deep learning for detecting robotic grasps," *The International Journal of Robotics Research*, vol. 34, pp. 705–724, April 2015.

[11] S. Kumra, S. Joshi, and F. Sahin, "Antipodal robotic grasping using generative residual convolutional neural network," in *International Conference on Intelligent Robots and Systems (IROS)*. IEEE, October 2020, pp. 9626–9633.

[12] J. Varley, J. Weisz, J. Weiss, and P. Allen, "Generating multi-fingered robotic grasps via deep learning," in *IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 4415–4420.

[13] J. Lundell, E. Corona, T. N. Le, F. Verdoja, P. Weinzaepfel, G. Rogez, F. Moreno-Noguer, and V. Kyrki, "Multi-fingan: Generative coarse-to-fine sampling of multi-finger grasps," in *IEEE International Conference on Robotics and Automation*, May 2021, pp. 4495–4501.

[14] Q. Lu, M. V. D. Merwe, B. Sundaralingam, and T. Hermans, "Multifingered grasp planning via inference in deep neural networks: Outperforming sampling by learning differentiable models," *IEEE Robotics and Automation Magazine*, vol. 27, no. 6, pp. 55–65, 2020.

[15] R. Newbury, M. Gu, L. Chumbley, A. Mousavian, C. Eppner, J. Leitner, J. Bohg, A. Morales, T. Asfour, D. Kragic *et al.*, "Deep learning approaches to grasp synthesis: A review," *IEEE Transactions on Robotics*, vol. 39, no. 5, pp. 3994–4015, 2023.

[16] F. J. Chu, R. Xu, and P. A. Vela, "Real-world multiobject, multigrasp detection," *IEEE Robotics and Automation Letters*, vol. 3, pp. 3355–3362, October 2018.

[17] K. Li, N. Baron, X. Zhang, and N. Rojas, "Efficientgrasp: A unified data-efficient learning to grasp method for multi-fingered robot hands," *IEEE Robotics and Automation Letters*, vol. 7, pp. 8619–8626, October 2022.

[18] S. Ainetter and F. Fraundorfer, "End-to-end trainable deep neural network for robotic grasp detection and semantic segmentation from rgb," in *International Conference on Robotics and Automation (ICRA)*, May 2021, pp. 13 452–13 458.

[19] J. Redmon and A. Angelova, "Real-time grasp detection using convolutional neural networks," in *International Conference on Robotics and Automation (ICRA)*, May 2015, pp. 1316–1322.

[20] H. Cheng, D. Ho, and M. Q.-H. Meng, "High accuracy and efficiency grasp pose detection scheme with dens predictions," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 3604–3610.

[21] Y. Wu, F. Zhang, and Y. Fu, "Real-time robotic multigrasp detection using anchor-free fully convolutional grasp detector," *IEEE Transactions on Industrial Electronics*, vol. 69, pp. 13 171–13 181, December 2022.

[22] A. Clark, M. Ciocarlie, and P. K. Allen, "The cornell grasping dataset," http://pr.cs.cornell.edu/grasping/, 2009, accessed: August 18, 2024.

[23] A. Depierre, E. Dellandrea, and L. Chen, "Jacquard: A large scale dataset for robotic grasp detection," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, October 2018, pp. 3511–3516.

[24] Kinova Inc, "Kinova kortex gen3 7-dof robot arm," accessed: August 18, 2024. [Online]. Available: https://www.kinovarobotics.com/product/gen3-robots

[25] SCHUNK Inc. Schunk dextrous hand 2.0 (sdh 2.0). http://www.schunk.com. Accessed: August 18, 2024.

[26] G. Jocher. (2020) Yolov5 by ultralytics. Accessed: August 18, 2024. [Online]. Available: https://github.com/ultralytics/yolov5

[27] P. D. D. R. Meneghetti *et al.*, "Annotated image dataset of household objects from the robofei@home team," October 2020.

[28] H. Zhang, M. Tian, G. Shao, J. Cheng, and J. Liu, "Target detection of forward-looking sonar image based on improved yolov5," *IEEE Access*, vol. 10, pp. 18 023–18 034, 2022.

[29] I. S. Isa, M. S. A. Rosli, U. K. Yusof, M. I. F. Maruzuki, and S. N. Sulaiman, "Optimizing the hyperparameter tuning of yolov5 for underwater detection," *IEEE Access*, vol. 10, pp. 52 818–52 831, 2022.