DEPARTMENT OF INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Biomedical Computing

6-DoF Grasp Learning in Partially Observable Cluttered Scenes

Dmitry Olefir

DEPARTMENT OF INFORMATICS

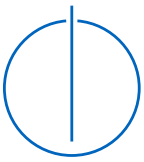
TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Biomedical Computing

6-DoF Grasp Learning in Partially Observable Cluttered Scenes

Lernen von 6-DoF Greiferposen mittels Neuronaler Netze in überfüllten und teilweise verdeckten Szenen

Author:	Dmitry Olefir
Supervisor:	PD Dr. habil. Rudolph Triebel
Advisor:	Martin Sundermeyer
Submission Date:	20.05.2021



I confirm that this master's thesis in biomedical computing is my own work and I have documented all sources and material used.

Munich, 20.05.2021 Dmitry Olefir

Abstract

The key element of the efficient interaction of an intelligent robot with its immediate environment is object manipulation - a task that current data-driven methods reshape into various methods aimed at object localization, classification, segmentation, and grasp pose estimation. This work is concerned with the grasp pose estimation, namely with the implications of 6-DoF grasp pose estimation for partially visible cluttered scenes.

In this thesis, two methods are proposed to address the problem of collision management of the grasp proposals and the full target scene due to the partial visibility and cluttered nature of a scene. The first explores the possibility of embedding input data with differential geometrical shape information, namely the modified mean curvature measure, to improve the qualitative results of grasp estimation. The second method proposes a supervisor network architecture termed Collision-GraspNet that classifies grasp proposals with respect to collision with the scene, including its occluded parts, and improves the invalid proposals via iterative pose sampling.

The first proposed approach is tested on the Contact-GraspNet model and compared with GraspNet architecture baseline performance. In its turn, Collision-GraspNet is compared with an analytical proposal filtering approach employed by GraspNet, and evaluated in three stages using various datasets.

Grasp supervisor architecture Collision-GraspNet outperformed the respective analytical approach and showed high confidence threshold flexibility. However, curvature-embedded data failed to improve upon the baseline model performance.

Contents

1 Introduction

Technological advancements of the last decades have paved the way for the massive automation, advances in medicine and the rise of Artificial Intelligence are transforming the world around us. When it comes to robotics, the fast pace of advancements in mechanics and electronics are pushing the boundaries of what is possible in terms of structural design, while researchers actively pursue the Holy Grail of robotics - making robots intelligent. This combined effort has changed the way we see and use robots: from bulky power-hungry manipulators performing primitive repetitive tasks on assembly lines to swarms of robots managing huge warehouses with little human intervention. However, the abilities of such robots are still limited, with limits being imposed by a multitude of factors. One such factor is the lack of data-driven methods for solving tasks robots face in complex real-world environments. While it is certainly true that serious progress has been made made in this field in the past 20 years, the final goal of turning a robot into a fully autonomous intelligent agent capable of solving complex tasks within dynamic environments is still unreachable.

One of the most promising approaches to achieving the aforementioned goal are the learning-based methods that allow a robot to learn from complex data to provide its systems with crucial parameters for planning and executing the tasks. In this regard, the task of robotic manipulation is exceptionally important. From the hardware standpoint, robots are already capable of performing high-precision movements, and a variety of end effectors and 3D sensors that become increasingly available reduces the effort of obtaining this data. This is paving the way towards seamless integration of robots into various industries, where they will be able to perform manipulation tasks semi-autonomously. Given the composition of sufficiently robust learning-based system and suitable hardware, robotic manipulators will no longer be restricted to static controllable environments like factories, but rather "invade" everyday spaces where they will be able to work

alongside humans. One example of robotic technology that has achieved such integration, albeit to a limited degree, is collaborative robots. They have become a proof that, given the right design of the components, robots can be incorporated into various businesses to boost the performance or production output, regardless of how small the production pipeline is. Yet they still require a highly controlled workspace and perform optimally in static environments.

In order to become truly efficient and autonomous at robotic manipulation task, a robotic system has to have an ability to reliably obtain 3D measurements of the target areas in space, localize and even classify the objects present in this areas, estimate and execute the best grasp pose according to some metric given the constraints imposed by the kinematics of the manipulator and structural complexity of the target area. Achieving such level of autonomy and awareness will allow such robotic systems to populate the places where historically only humans could function. Consider an operating room - a facility found withing virtually any modern hospital. The vast majority of surgical operations taking place in such an environment requires a team of highly trained professionals, each required to show exceptional precision and concentration during the surgeries that may span hours upon hours. The aforementioned intelligent robot could alleviate the workload imposed on the staff during the surgery by performing tasks such as disinfecting the tools and supplying them to the surgeon, and even operating some of the equipment. This would allow to not only potentially reduce the team size (and consequentially the cost of surgeries), but also reduce the probability of a human-induced mistake.

Such integration based on effective control of robots in their surroundings is not yet feasible for many scenarios, but the researchers are actively investigating the ways of learning from 3D measurement data to imbue robots with relevant experience. Such 3D information aims to capture the geometry of the environment and can take many forms. However, within the scope of this thesis the following representations were handled.

1.1 Structured data

The 3D data representations described below have an underlying Euclidean structure. In particular, the properties of the data structured on a grid such as having a global

parametrization and a common coordinate system are preserved. The following representations are examples of structured data:

RGB-D images. Representing 3D data as RGB-D images is becoming increasingly popular due to the rising availability of RGB-D sensors, e.g. Kinect by Microsoft, or RealSense from Intel. Such representation provides a 2.5D information about captured area by obtaining color information (RGB) along with a depth map (D). This data representation is illustrated in Figure 1.1.

Figure 1.1: RGB (left panel) and depth (right panel) images from the GraspNet-1Billion dataset [Fan+20].

Voxel grids. Voxel grids represent 3D data as a regular grid in the three-dimensional space and allow to model 3D data by describing how the 3D object is distributed through the scene. Viewpoint information can be encoded via labeling the voxels that are occupied as visible, occluded, or self-occluded. It must be noted, that the applicability of voxel grids for modeling of high-resolution data is severely limited since the representation always describes both occupied and non-occupied parts of the scene, thus enforcing enormous requirements in regards to memory storage. This data representation is shown in Figure 1.2.

1.2 Non-structured data

The following 3D data representations are the non-structured representations.

3D point clouds. This representation describes 3D data as a set of unstructured 3D points that approximate the geometry of 3D objects. Absence of the connectivity information leads to ambiguity of the surface information. This data representation is illustrated in Figure 1.3.

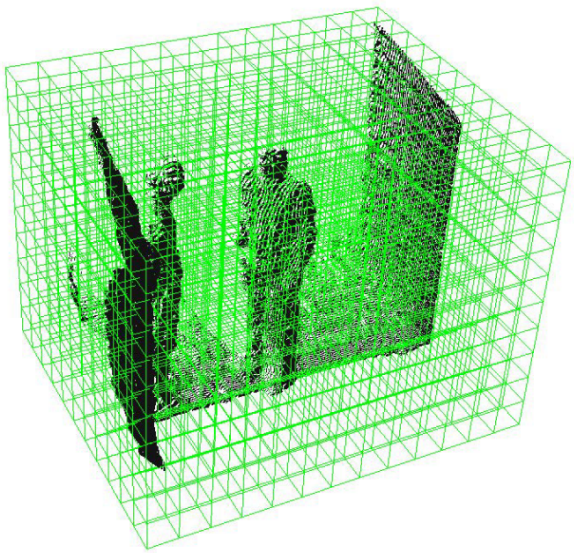


Figure 1.2: Voxel occupancy representation of a 3D structure according to [Wan+12].

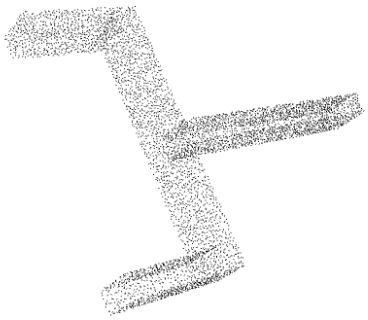


Figure 1.3: 3D point cloud of a fork gripper model. Contains 5000 points.

Meshes. Meshes represent 3D data as a set of polygons (faces) that are described in terms of a set of vertices that define the 3D coordinate of the mesh in space. It is one of the most popular representations of 3D shapes. Vertices are associated with a connectivity list which describes how these vertices are connected to each

other. Geometry of the meshes can be characterized as a subset of the Euclidean space following the grid-structured data. This data representation is shown in Figure 1.4.

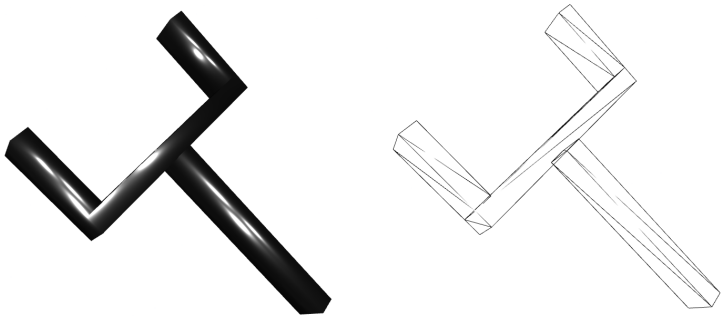


Figure 1.4: Mesh of a simplified fork gripper: the gripper with the visualized faces (left panel); the gripper with only the wireframe of the mesh visualized (right panel). Mesh contains 32 vertices and 32 faces.

Various representation allow for a multitude of approaches to explore the possible solutions for a number of sub-tasks that constitute the problem of vision-based robotic grasping.

1.3 Problem Statement

In this thesis arguably the most important and difficult sub-task of the vision-based robotic grasping is explored, namely 6-DoF grasp estimation in partially observed cluttered scenes, as it has to deal and address numerous limitations imposed, such as high structural complexity of the scene and quite limited available 3D information about the scene.

Grasp estimation on the partially visible scene involves a lot of reasoning about the occluded space. It means that those grasps that are predicted for an object that is partially visible can be invalid during the execution of the grasp due to the, for example, predicted grasp pose leading to the collision with the occluded part of the object. In this thesis two possible way of alleviating this constraint are proposed. First proposes using data embedded with differential geometrical

scene information during learning and inference of a chosen baseline architecture *Contact-GraspNet*, in order to provide information about possible shape of the occluded parts of the target scene. Second, in its turn, proposes an architecture *Collision-GraspNet* that is capable of reliable classification and modification of colliding grasp pose proposals using the same partially observed scene, the gripper point cloud representation, and a proposed grasp pose as an input.

1.4 Thesis Structure

This thesis is structured as follows. Chapter 1 describes the motivation and provides a problem statement. In Chapter 2, a brief overview of the existing approaches to the various vision-based robotic grasping tasks is given. Chapter 3 presents an overview of publicly available datasets and of the two datasets that are extensively used throughout the work. Chapter 4 relays the core methods and approaches tested in this work. Chapter 5 provides an overview of the conducted experiments as well as the analysis of the obtained results. Future steps are discussed in chapter 6. Finally, the outcomes of work presented hereby is summarized in chapter 7.

2 Related work

The aim of this chapter is to provide a brief overview of the available Vision-based Robotic grasping methods. The first section concerns classical computer vision approaches. In the second section, the attention is shifted towards learning-based methods, comparing end-to-end and modular approaches. The third section considers specifically the task of grasp estimation in two cases: 2D planar and 6-DoF grasp estimation. The fourth section highlights the methods of 6-DoF grasp estimation on complete and partial shapes, while the closing section lists the differences of the grasp estimation of single objects versus cluttered scenes.

2.1 Classical computer vision approaches

As a fundamental robotic manipulation task, robotic grasping is well-defined and researched. Up to the beginning of the current millennium, the field was dominated by the analytical approaches. Classical computer vision methods explored the main components of what would compose the vision-based robotic grasping field and developed a multitude of methods that can be roughly classified based upon the specific subtask of robotic grasping they consider as follows:

Object localization methods employ methods of shape primitives like ellipses and polygons fitting in 2D [FF+96]; [DP73] and localization using 3D primitives like in RANdomSAmple Consensus (RANSAC) [FB81], or Hough-like voting methods [RVDH05]. **Salient object detection methods** in 2D [Jia+13] and 3D [Pen+14] were also proposed and investigated. **Object pose estimation methods** were also considered in the classical computer vision research. In various works, both 2D cases using hand-crafted global and local descriptors [Low99]; [RD05]; [BTVG06]; [Rub+11], and 3D cases [Joh97]; [Fro+04]; [RBB09]; [Ald+11] were investigated. The topics of **2D planar grasp** [Dom+14]; [JMS11a]; [VPB19] and of **6-DoF grasp** [Mil+03]; [BK10]; [PP15] **pose estimation** were also researched.

As it is usually the case with such methods, they required various often unrealistic assumptions, such as sensor models being noiseless, models of the robot kinematics and dynamics being perfect, or having the object model and it's relative alignment in regards to the manipulator available [MLS94]; [Shi96]. Such assumptions resulted in the limited applicability of the mentioned approaches in realistic scenarios, especially in the dynamically changing environments.

2.2 Learning-based approaches

In the last decade, the advancements in the 3D data acquisition hardware and machine learning algorithms lead to data-driven approaches taking the lead in the field of robotic grasping. The main divide that delineates the learning based approaches in the field lies in the way the grasping task is treated. On one hand, it can be thought of as a single monolithic pipeline that the data-driven method tries to implement as a whole. The approaches that treat the problem in such a manner are called end-to-end. On the other hand, modular approaches constitute a perhaps more flexible alternative by breaking the global task of grasping into components and trying to build algorithms that solve the sub-tasks related to just a single such module. A more detailed review of the discussed methodologies is presented in the following subsection.

2.2.1 End-to-end learning

Treating the complex task of grasping as a single pipeline of perception, planning and execution produces monolithic system is mostly characteristic of reinforcement learning (RL) based methods. Learning grasping directly from self-supervision of a system's interaction with the environment allows such a system to become progressively better at grasping though the repeated experience, and, possibly, to achieve relatively high degree of efficiency without any substantial human intervention or control.

While supervised learning for grasping setups usually uses large (thousands to millions) amounts of grasp examples for hundreds of different objects, for RL methods this regime is troublesome: assuming learning is primarily on-policy, the system has to revisit already seen objects again and again to avoid forgetting

them, which makes proper generalization to diverse sets of novel objects quite hard. Off-policy RL methods are a preferred choice for tasks such as grasping, where the wide variety of previously seen objects is crucial for generalization. Recent research on deep Q-networks enables designing closed-loop grasping strategies. Some demonstrate this by proposing a Q-function optimization technique to provide a semi-scalable approach for vision-based robotic manipulation applications [D.+18], while others use deep Q-learning for learning grasping strategies, for applications in limited spaces and cluttered target environments [A.+18], though those are often limited to execution on single and/or basic geometric target objects [IP17]. Main disadvantages are constituted by the notorious instability of RL, which manifests in the difficulties during tuning. In theory, this step is not necessary, though the reward function is often manually tuned regardless [RIP17]. End-to-end learning strategy of joint perception and execution leads to reduced generalization to novel objects. Moreover, the trial and error approach in collision-sensitive and highly structured real environments is far from being an easily available option for a real robotic setup due to the danger of damaging the said environment or expensive robotic system, unless carried out in simulation.

2.2.2 Modular learning

Outside of the RL framework, the vision-based robotic grasping task is divided into three key tasks instead of treating it as a single pipeline. Those key tasks are explored separately, and the robotic grasping system itself is often viewed as a complex system composed of the following sub-modules:

- the grasp detection system;
- the grasp planning system;
- the control/execution system.

The grasp detection system, the primary task of which is to determine the way to grasp an object, is often viewed the key component in the pipeline and is being actively researched. Such system is shown in Figure 2.1 in a simplified manner. As shown in 2.1, the grasp detection task can be broken down into the three main subtasks which are discussed in the following subsections: object localization

(determining where the object is located in 3D space), object pose estimation (determining the orientation of the object in 3D space) and grasp estimation (determining the grasp parameters).

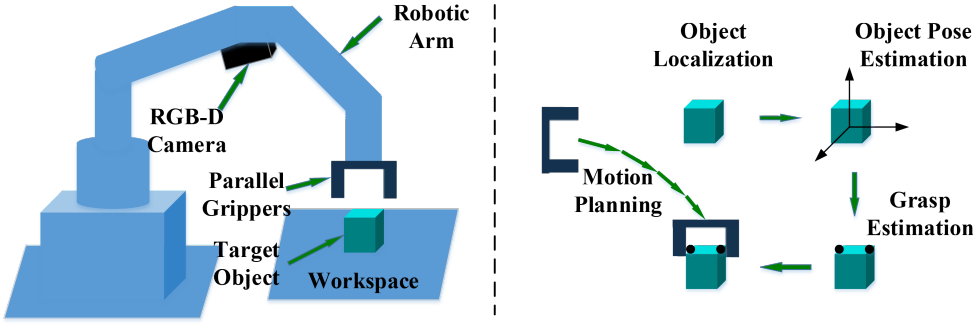


Figure 2.1: The grasp detection system according to [Du+20]: (left panel) robotic arm, depth sensor, parallel gripper, and a target object placed on a planar work surface. (right panel) Grasp detection system composed of target object localization system, object pose estimation system, and grasp estimation system.

2.2.2.1 Object localization

As mentioned above, the first task in the grasping pipeline is that of object localization. It is usually can be treated as one of three following cases.

In the case of **object localization without classification**, the system outputs the approximate regions of the objects without knowing their categories. This case is typically considered in grasp-related robotic tasks, for example, in industrial environments, where objects present have fixed shapes. According data-driven solutions exist for 2D cases [Zha+15]; [Zha+16]; [LH16]; [LHY18]; [Lin20]; [Qi+19], and 3D RGB-D cases [Pia+19]; [Pan+20].

The second case is **object detection with classification**, where model provides bounding boxes of the target objects in the scene as well as their categories. Both 2D and 3D cases are represented by two-stage and one-stage methods. Two-stage methods first propose candidate objects and select the most suitable candidate in the second step. One-stage models skip the region-proposal step. For 2D case several two-stage [Gir+14]; [Gir15]; [Ren+15]; [Dai+16]; [Lin+17] and one-stage

[Red+16]; [RF17]; [RF18]; [BWL20] methods exist. Same is true for two-stage [Qi+18]; [XAJ18]; [Lia+19b]; [Shi+20]; [SWL19]; [Xie+20b] and one-stage [ZT18]; [YML18]; [JMS11a]; [SR20]; [Naj+20] methods for 3D.

While 2D object detection provides sufficient information for the execution of 2D planar robotic grasping tasks, mere detection in 3D does is not sufficient for the actual execution of a grasp. However, the estimated 3D bounding boxes could prove useful for collision avoidance.

The third case is **object instance segmentation** where the task is to output the pixel- or point- regions of the target objects along with their categories. The general idea is to provide specific regions of the target object described by the input data. Yet again, there are several two-stage [He+17]; [Che+18]; [Fan+20] and one-stage [Xie+20a]; [Che+19]; [Tia+19b]; [Che+20] methods for 2D case, as well as for 3D [Yi+19]; [HDN19]; [Eng+20]; [LF19]; [Zha+20a]; [Han+20].

2D and 3D object instance segmentation are highly important problems in robotic grasping setups. 2D cases are already widely used in respective tasks, while segmentation on 3D input still relies on the segmentation of RGB component of RGB-D input. 3D object instance segmentation develops quite rapidly as a research field, thus the degree of its integration into robotic grasping relies on the improvement of the corresponding methods in terms of speed and performance.

2.2.2.2 Object pose estimation

The second main task is object pose estimation, where the general idea is to estimate the 6D object pose to allow the generation of grasp poses for the already known objects. The methods that attempt to solve this task can be broadly categorized as follows.

Correspondence-based methods utilize deep 3D descriptors for matching 3D points that are representative and discriminative [MSG20]; [Zen+17a]; [SSH20]; [Hu+19]; [HBM20].

Template-based methods are aimed at accomplishing the partial registration task via taking a pair of point clouds as an input and extracting discriminative features using 3D deep networks with later regression of the relative transformations between each point cloud [GGF20]; [Sar+19]; [WS19a]; [Aok+19]; [WS19b]. Some methods learn orientation implicitly [Sun+18], while others directly estimate the

6D pose from the input image [Liu+19].

In **voting-based methods**, each input 3D point contributes towards the estimation of the 6D object pose via providing one or more votes within a direct [Wan+19] or indirect [Yu+20]; [Wan+20] voting paradigm.

2.2.2.3 Grasp estimation

The third main task, which is the focus of this work, is grasp estimation, which allows estimating a gripper pose in the camera coordinate system. Grasp estimation implicitly includes all the previous tasks, and estimates the poses directly on the input target scene. This task exists in two setups: 2D planar grasp estimation, where the grasp is constrained from one direction and the grasp pose is reduced to a 3D pose; and 6-DoF estimation, where the gripper can approach and interact with the target objects from various directions. Both cases are discussed in detail in the following sections.

2.3 2D vs. 6-DoF grasp estimation

During the times when capturing depth data was not a trivial task, the field of grasp estimation was mainly concerned with planar grasp estimation. 2D planar grasps are constrained from one directions and lie in a planar workspace, thus the information is reduced from 6 to only 3 degrees of freedom (2D in-plane position plus 1D gripper rotation angle). Oriented rectangles are the go-to configuration description for planar grasps that define them uniquely [Lev+16]; [PG15]; [MG17]; [Zen+18]. Methods for 2D planar grasping can be categorized into methods of grasp contact points evaluation and methods of oriented rectangles evaluation.

Grasp contact points in 2D planar case uniquely define the gripper's pose, and the corresponding data-driven methods realize the estimation of the most probable grasp contact points via pixel-wise grasp affordances. Some of those methods are concerned with the prediction of pixel-wise affordances with respect to the set of grasping primitive actions. Such methods generate grasp qualities for each pixel of the input image, and then the set of points with the highest value of affordance is to be executed [Zen+18]; [Cai+19]. This can be achieved, for example, via inferring dense affordance probability maps for grasping primitive actions by using

fully convolutional networks [Zen+18], or via learning antipodal grasps by using fully convolutional residual networks [Cai+19]. A slightly different take on this is Generative Grasping Convolutional Neural Network which predicts pixel-wise grasp quality and pose [MCL18].

Methods of oriented rectangle evaluation exploit the fact that 2D oriented rectangles also define a unique gripper pose. In this case, deep learning methods are divided into three categories: classification-, regression-, and detection-based methods. Classification-based methods train classifiers to evaluate candidate grasps, and select the one with the highest assigned score [LLS15]; [PG16]. Regression-based methods train a model to provide grasp location and orientation directly [RA15]; [Zha+19]. Detection-based methods use the reference anchor box, to assist the generation and evaluation of grasps [DDC20]; [CXV18].

In general, the main drawback of a planar grasp representation is defined by the fact that such representation limits the diversity of the outputted grasps: grasping of an arbitrary object might be simply impossible given the imposed constraints. This effectively limits the applicability of 2D grasp estimation methods in many real world scenarios. On the other hand, estimation in 3D with 6-DoF is not subject to such constraints. The difference between the two approaches is visualized in Figure 2.2, where 6-DoF ground truth grasps from *GraspNet-1Billion* dataset [Fan+20] are converted into planar grasps, resulting in a drastic loss of variability.

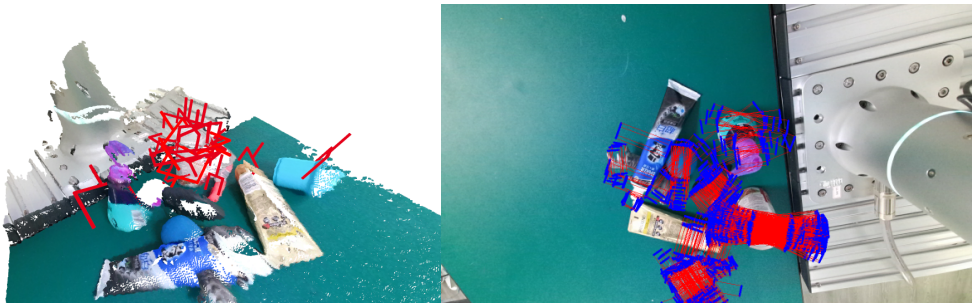


Figure 2.2: 6-DoF (left panel) to 2D planar (right panel) grasp conversion using *GraspNet-1Billion* dataset's ground truth grasps.

For 6-DoF grasping, the gripper has more options when accessing the object: the additional degrees of freedom allow for a much more variable pose of the gripper, which is an essential component to execute the grasping. A detailed review of the

available 6-DoF grasping methods is presented in the following section.

2.4 Grasping on complete vs. partial shapes

As mentioned above, the 6-DoF grasping provides more flexibility when solving the task of robotic grasping making it an advantageous alternative to the 2D approaches. 6-DoF grasp estimation methods can be divided into two groups: methods based on the complete shape and methods based on the partial shape. Both groups are discussed in the following subsections.

2.4.1 Grasping on complete shapes

6-DoF grasp estimation methods on complete shapes are useful for cases when obtaining full scene information is possible, for example via explanatory movements or via multi-view fusions. Subsequently, the poses of the known objects can be estimated and the corresponding grasp poses can be obtained on the complete 3D shape. The resulting 6-DoF grasp poses can be subsequently transformed from the object coordinate system to the camera frame. Additionally, a complete shape of the scene or object can be reconstructed from the single-view point cloud, and then used for grasp pose estimation.

In the case of the estimation using the full shape, many deep learning methods are utilized as tools for assisting the robotic grasp tasks. Some methods, such as [Zen+17b], estimate the poses through partial registrations, where multiple views of a scene are segmented and label with a fully convolutional neural network. The object models that were scanned prior are placed into the segmentation result to produce their poses. In contrast, there are methods that achieve object and grasp pose estimation in a joint fashion using convolutional neural network pipelines, such as SilhoNet [BJR19]. Some methods utilize RGB-D data for object segmentation and partial registration to obtain object poses, from which suitable grasp poses can be computed with high accuracy [Won+17]. DenseFusion [Wan+19] demonstrates high success rate in assisting with practical robot grasping tasks by proposing a heterogeneous architecture that processes RGB and depth data sources individually and extracts pixel-wise dense feature embedding with consequential iterative pose refinement.

Naturally, the applicability of the aforementioned methods is determined by the availability of the full scene information. Obtaining such information poses a separate problem for which a multitude of solutions have been proposed. For instance, the methods that employ shape completion in order to estimate grasps aim at solving this challenge by producing the complete geometry from the partial observations, thus making the estimation of grasp poses more precise.

Among such approaches there are those that reconstruct the geometry from the partial point cloud. For example, the general idea proposed in [Var+17] is enabling the robotic grasping via shape reconstruction by employing a 3D convolutional neural network (CNN) to actually complete the shape. For the objects that are to be grasped, a high-polygon mesh is generated, with a low-polygon mesh generated for the rest of the scene. The resulting grasps are evaluated using GraspIt! framework [MA04]. [LVK19] proposes a deep neural network shape reconstruction architecture, and a probabilistic planning method that uses the shape uncertainties highlighted by the reconstructor network to produce grasps. PointSDF by [Mer+20] trains a continuous signed distance function embedding for the partial object point cloud, and a grasp success model in simulation, enabling geometric awareness in a grasping system. [Tos+20] uses two networks, first of which produces grasp proposals, while the second employs 3D shape reconstruction to refine the candidate grasps.

In contrast to the aforementioned methods that work with partial point clouds, there are approaches that perform reconstruction using single-view RGB(-D) data. [Wan+18] learns shape priors the predict the object's shape from a single RGB image. Such priors are learned from large-scale shape repositories and the incorporated tactile sensing observations are subsequently used to refine the shape. Deep Geometry-aware Grasping Network (DGGN) [Yan+18] produces geometry-aware grasps by learning 6-DoF grasp poses from RGB-D data and a complete shape produced by shape generation network. ClearGrasp [Saj+20] tackles the problem of estimating the geometry of the transparent objects by using a single RGB-D image and deep convolutional networks to intersurface normals, masks of transparent surfaces, and occlusion boundaries, that are then used to refine the initial depth estimates.

In general, the main drawback of the methods that employ shape completion stems from the fact, that grasping accuracy directly depends on the accuracy of

shape completion, which in turn is challenging to generalize properly to novel shapes. When using single-view data, the lack of information about the geometry of the object which is not visible to the camera from a single given pose, has a critical effect on the completion accuracy.

2.4.2 Grasping on partial shapes

The methods summarized herein do not require complete shapes to estimate grasping, making them more flexible as compared to the counterparts that do require such information. One group of methods that operate on partial shapes transfers grasps from already known existing shapes. This means that the methods in this group try to find some sort of correspondence between the obtained partial data and some existing complete data that is available a priori. This typically happens after a classification step if it is ruled that both are of the same category. If the object is deemed similar to those that are in the database, correspondence-based methods can transfer the grasp points from complete 3D object to partial-view object. [Tia+19a] proposes a method that assumes the similarity of topology and shapes between the example and the novel objects in order to produce grasp configurations. Active learning approach produces grasp contact mapping between the known and novel objects based on the 3D segmentation of the novel object that takes into account geometrical and semantics information. In its turn, DenseObject Net [FMT18] employs a self-supervised approach to learn the dense descriptors that are used as a representation for the robotic manipulation, which enables manipulation of deformed objects and manipulation in clutter. Dense Geometrical Correspondence Matching network (DGCM-Net) [PPV20] utilizes metric learning to encode geometrically similar objects into a feature space to obtain relevant experience for novel object via nearest neighbor search.

In contrast to the aforementioned methods, following methods compose a group that operate directly on partial geometrical data, namely partial point clouds. GPD [Pas+17b] produces candidate grasps in a region of interest that are encoded in a stacked multi-channel image, where each candidate is assigned a score via evaluation with a convolutional network. [LYC20] generates candidate grasps via uniform sampling over the whole 3D space, to predict the grasp quality and reachability using 3D convolutional network. PointnetGPD [Lia+19a] produces candidate grasps

via random sampling and evaluates them by direct analysis using 3D deep neural network PointNet [Qi+17b] backbone. Ground truth grasps are labeled using the force-closure metric. 6-DoF Grasp-Net [MEF19a] employs Variational Auto-encoder to sample candidate grasps to refine them with an evaluator network. For both steps PointNet++ backbone [Qi+17a] is used. [Mur+20] can be viewed as a direct improvement of 6-DoF Grasp-Net, that additionally learns a collision checker model that uses the raw point cloud of the scene and of the gripper model. Collision checker refines the rejected grasp candidates using Metropolis-Hastings sampling that yields similar performance to the gradient-based one while it is computationally twice as fast. S4G [Qin+20] model is based on the single-shot proposal network with Pointnet++ backbone that produces amodal grasp proposals using regression directly in contrast to 6-DoF Grasp-net that uses encoding and decoding. REGNet [Zha+20b] is a three-stage network composed of Score Network that selects positive points with high grasp confidence, Grasp Refine network that generates grasp proposals on these points, and Refine network that refines the grasps based on local features.

GraspNet [Fan+20] learns the approaching direction and operation parameters of grasps in a decoupled manner with an additional grasp affinity field that is designed to improve the grasping robustness. *Contact-Graspnet* [MS21] is an end-to-end generative network that estimates grasps directly in cluttered scenes. It employs a novel grasp representation that includes the contact points of the parallel jaw gripper.

Due to the inherent flexibility of 6-DoF grasp pose estimation and its applicability to various real-world scenarios, it considered and exclusively used in this thesis. Furthermore, since the majority of methods work with point clouds, this work considers point cloud data representation as well.

2.5 Grasping in cluttered scenes

Learning-based methods for grasping are mainly focused on dealing with isolated objects on planar surfaces [MEF19a]. This work, however, is focused on object grasping in a cluttered scenario. This problem is significantly more complex due to the restricted access to the objects, since the immediate environment is occupied

by other objects, and the usable workspace thus severely limited. Cluttered setup can also be described by two distinct situations.

The **bin-picking case** is characterized by the scene's object being located in a stable pile. As an example one may imagine a case of an industrial bin-picking scenario, where a literal bin is filled with parts and objects that require picking up. Such case is shown in the Figure 2.3. Here a lot of geometry-related information is occluded by the neighboring objects. Collisions imposed by the executed grasps are not critical since the objects are already in a pile.

Figure 2.3: Examples of a bin-picking scene. Taken from [MG17]

In contrast, in the **case of structured clutter**, the scene is characterized by a set of tightly packed objects of different sizes. A descriptive example is a stocked shelf of a kitchen's pantry or of a supermarket. In comparison to bin-picking, structured clutter does not allow for dense distribution of grasps, since any, even minor unintended collision may lead to serious consequences and failure of the task. Objects in structured clutter have less stable safe poses since they are not combined in a pile, and collision avoidance becomes exceptionally important. An example of such setup is shown in Figure 2.4. Additionally, occlusions imposed by other objects are not only more likely to happen, but also restrict the possibilities of perception, and in combination with possible contact interactions between objects affect the predictions quality. Thus, acquiring high-quality information may require, for example, using multiple views that in their turn would require gathering via exploration, which is not possible in confined spaces.

Figure 2.4: Examples of simple structured clutter scenes. Taken from [Mur+20]

As an example of a real world application, one may imagine a case where a robot is ordered to pick up a heavy glass jar from the aforementioned pantry. Even if the predicted gripper pose is not leading to direct collision, executing such a grasp can be hard, since the planned path must be collision free with the environment and still kinematically possible (given the manipulator configuration and constraints) for the robot at the same time. Some methods that tackle the problem of grasping in structured clutter exist. In the scope of this thesis two solutions are used, namely *GraspNet* [Fan+20] and *Contact-GraspNet* [MS21]. Former one proposes an end-to-end grasp pose prediction network given point cloud inputs, where approaching direction and operation parameters are learned in a decoupled manner. Additionally, *GraspNet* employs a *Model-Free Collision Manager (MFCM)* module that analytically filters the predicted grasp poses on the grounds of them colliding with a voxelized input cloud. *Contact-GraspNet* proposes an end-to-end generative grasp network to predict a distribution of 6-DoF grasps directly in cluttered scenes using novel grasp representation, which includes the contact points of the parallel jaw gripper, thus the 3D points in an input point cloud are viewed as potential grasp contacts. The proposed method reduces the dimensionality for grasp representation to 4-DoF via rooting of the full 6-DoF

grasp pose in the recorded scene point cloud, which additionally includes predicted grasp with. During inference local regions of interest can be optionally extracted around the 3D centroid of point cloud segments in order to maximize the number of potential contact points.

None of the aforementioned approaches address the limitations imposed by partial visibility of the scene directly during training. In this thesis two ways of addressing the limited geometrical information are proposed: first aims at employing differential geometrical information to embed training and testing data in order to qualitatively improve predictions, while the second addresses the problem of collisions in a shape of a new architecture *Collision-GraspNet* that aims to reliably classify proposal grasps as colliding or valid using the same partially visible scene, and to modify colliding proposals to bring them out of collision while keeping them suitable for actual grasp execution.

3 Datasets

Datasets are the key to the success of learning-based methods. Following sections provide an overview of some of the publicly available datasets and of the two datasets that are extensively used in this work, namely *GraspNet-1Billion* and *ACRONYM*.

3.1 Overview

In terms of data-driven grasping, publicly available datasets differentiate in, among other parameters, the type of the provided observations, grasp label nature, and the amount of variability the dataset presents. Table 3.1 provides a brief comparison of the publicly available datasets for grasping.

It was already mentioned, that a significant portion of learning-based methods are concerned with predicting planar grasps, which are represented by rectangular grasps [JMS11b]; [Lev+16]; [DDC18]; [Zha+19]. Some datasets, however, provide grasp 6-DoF poses [KBS15]; [Mah+17]; [VMT17]; [EMF20]. Additionally, there is a choice between providing real measurements in RGB-D form [Lev+16]; [JMS11b]; [Zha+19] or synthetic ones in the form of depth images [KBS15]; [Mah+17] or RGB-D [DDC18]; [VMT17]. Arguably, the key difference between the datasets is the way the grasp labels are computed: real robot executions [Lev+16], analytical computation [Fan+20]; [Mah+17], or physics simulation [DDC18]; [KBS15]; [VMT17]. In the majority of cases, datasets provide grasps for isolated objects [JMS11b]; [DDC18]; [MEF19a], rather then for groups of objects that make up a cluttered scene [Fan+20]; [Lev+16]; [Zha+19]. In some cases, grasps from isolated objects can be reused in cluttered scenes, discarding those that collide [EMF20]. The last, but not the least important key factor, is the quantity of grasps provided in the dataset. Some datasets provide a variety of objects [Lev+16]; [Zha+19], while some contain a lot of grasps for a limited selection of objects [EMF19]; [Fan+20]. There is no

clear evidence regarding what is more important for successful grasp generalization, therefore in some cases datasets attempt to balance object-to-grasp ratio [EMF20]. It must be noted, that higher geometric (i.e. object) variability can be achieved by random shape generation [Tob+18].

3.2 GraspNet-1Billion

GraspNet-1Billion [Fan+20] is a large-scale dataset providing richly and densely annotated cluttered scenes. Figure 3.1 shows an overview of the key components of the dataset.

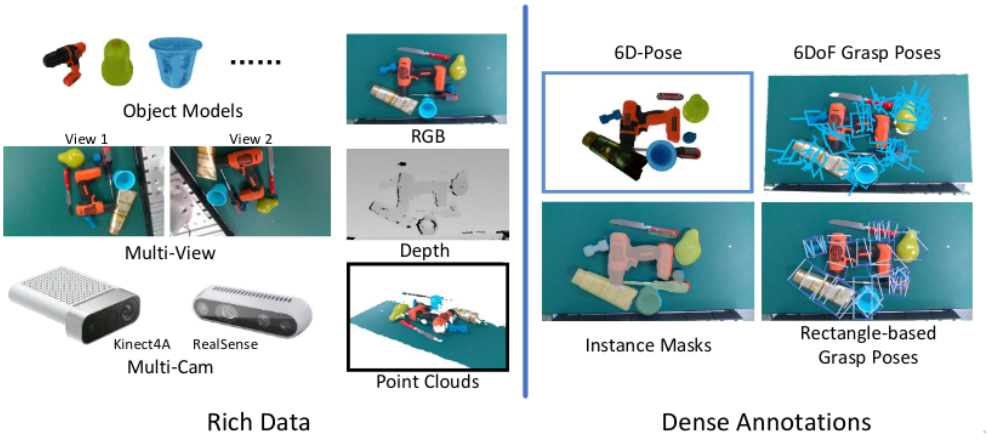


Figure 3.1: The key components of *GraspNet-1Billion* dataset. RGB-D images are taken using both RealSense camera and Kinect camera from different views. The 6D pose of each object, the grasp poses, the rectangle grasp poses and the instance masks are annotated. Taken from [Fan+20].

The dataset contains 88 objects with high quality 3D mesh models. 32 objects that are suitable for grasping are selected from the YCB dataset [Cal+17], 13 adversarial objects are selected from DexNet2.0 [Mah+17] and 43 more objects are unique to *GraspNet-1Billion*. This ensures geometrical diversity of the dataset.

For cluttered scene data collection, two popular RGB-D cameras, namely Intel RealSense 435 and Kinect 4 Azure, were chosen to simultaneously capture the scene. $8 - 12$ objects were randomly selected for each scene, where they were placed in a cluttered manner. The robot arm with cameras moved along a fixed trajectory that

covered 256 distinct view-points on a quarter sphere and a synchronized image pair from both RGB-D cameras as well as camera poses were saved for each view-point. Totaling 190 scenes, *GraspNet-1Billion* contains 48640 images per camera type. 100 scenes are used for training, the remaining 90 - for evaluation. These 90 scenes are split in 3 groups: 30 scenes composed of objects present in the training data (seen data), 30 scenes composed of objects similar to those in the training scenes (similar data), and 30 scenes composed of novel objects (novel data). The amount of grasp poses for each scene varies from $3,000,000$ to $9,000,000$, and in total the dataset contains around 1.1 billion grasp poses. Additionally, rectangle based grasp poses, object masks and bounding boxes are provided. Each frame is also associated with a camera pose, thus multi-view point cloud can be fused.

Grasp poses are annotated using an analytical computation method. *GraspNet-1Billion* adopts an improved [MEF19b] force-closure metric [Ngu88]; [Pas+17a]: given a grasp pose, the associated object and the friction coefficient μ, force-closure metric outputs a binary label indicating whether the grasp is antipodal under that coefficient. With $\Delta\mu = 0.1$ as interval, the coefficient decreased gradually from 1 to 0.1 step by step until the grasp is not antipodal anymore. The grasps are assigned a score that lies in $(0, 1]$ range, such that a grasp with a lower friction coefficient μ has a higher probability of success. Grasps are projected to the corresponding objects based on the annotated 6D object poses.

3.3 ACRONYM

ACRONYM [EMF20] is a dataset for robot grasp planning based on physics simulation. Figure 3.2 provides an example of geometric variety and grasp density provided by the dataset.

ShapeNetSem [SCH15] dataset is used as a source for object meshes. Meshes that contain more than one connected component are excluded, the rest is made watertight [HSG18] to ensure correct behavior during physics simulation for which uniform density is also assumed. In total, *ACRONYM* contains 8872 objects from 262 different categories.

The dataset focuses on parallel-jaw grippers ($DOF = 1$), specifically it uses the model of the Franka Panda gripper. When generating grasp proposals, it is assumed

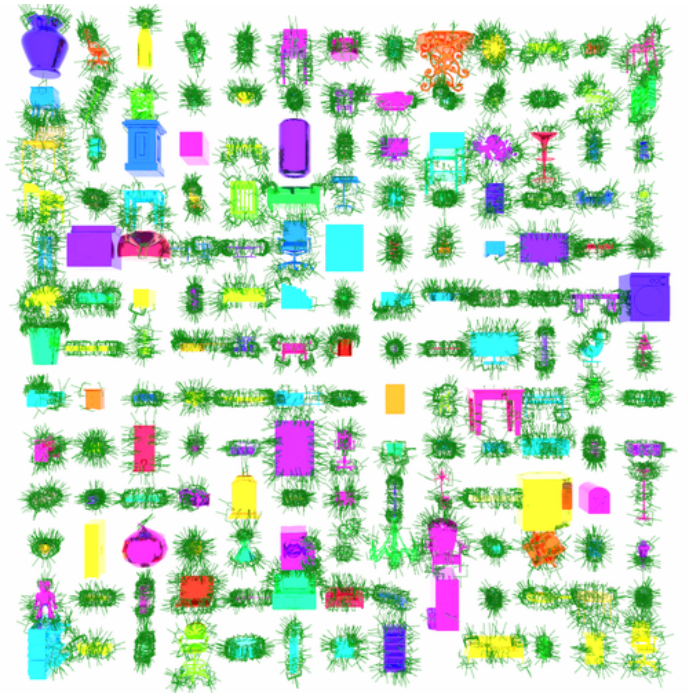


Figure 3.2: *ACRONYM* contains 2000 parallel-jaw grasps for 8872 objects from 262 categories, totaling 17.7M grasps. Taken from [[EMF20]].

that grasps are unsuccessful if an object and the gripper collide or if the intersection between the object and the volume between the gripper fingers is empty. For each object, 2000 grasp proposals are generated that pass this check during rejection sampling, which in uses an antipodal sampling scheme. *ACRONYM* accumulates $17.7M$ grasps in such a manner.

FleX physics simulator [Mac+14] is used to evaluate and label each grasp. Objects, gripper palm and fingers are simulated as rigid bodies. In addition to isolated objects, *ACRONYM* provides a mechanism to procedurally generate scenes with structured clutter. These scenes are generated by sampling a support object. To label such scenes with grasps, the grasp labels for isolated objects are reused, and the grasps that would collide with any scene geometry are labeled as failures. To obtain observations for generated scenes, *ACRONYM* provides an interface to render depth images, segmentation masks, and point clouds.

Dataset	6-DoF	Observ-s	Labels	Grasps	Objects (categories)	Grasps per object/scene	Scenes
Cornell [JMS11b]	No	Real	Manual	$8k$	240	33	Single
Jaquard [DDC18]	No	Sim	Phys. sim.	$1.1M$	$11k$	100	Single
VMRD+ [Zha+19]	No	Real	Manual	$100k$	$15K(31)$	≈ 6.5	Multi
Levine [Lev+16]	No	Real	Real-world	$650k$	N/A	N/A	Bin
Kappler [KBS15]	Yes	Sim	Phys. sim.	$300k$	700(80)	≈ 430	Single
Dex-Net [Mah+17]	Yes	Sim	Analytical	$6.7M$	1500(50)	100	N/A
Veres [VMT17]	Yes	Sim	Phys. sim.	$50k$	N/A (64)	N/A	Single
6-DoF GraspNet [MEF19a]	Yes	Sim	Phys. sim.	$7.07M$	206(6)	$43k$	Single
Eppner [EMF19]	Yes	N/A	Phys. sim.	$1B$	21	$47.8M$	Single
GraspNet-1Billion [Fan+20]	Yes	Real	Analytical	$1.1B$	88	$12.5M$	Multi
ACRONYM [EMF20]	Yes	Sim	Phys. sim.	$17.7M$	8872(262)	$2k$	Multi

Table 3.1: Comparison of publicly available grasp datasets: name of the dataset with a reference, grasp type, observation type (real or simulated), label type (manual, physics simulation, analytical), total number of grasps, number of objects (categories), number of grasps per object/scene, and scene type.

4 Method

Grasp estimation in partially observable structured cluttered scenes, which was discussed in detail in sections 2.3 and 2.4, involves reasoning on the occluded parts of the scene. Additional limitations imposed by the structured clutter nature of the observed scene make estimation of grasps that are viable in real setups more challenging. For example, estimation of the grasps that are not in collision with the occluded parts of the scene becomes quite important.

Firstly, two architectures concerned with 6-DoF grasp pose estimation and a corresponding evaluation pipeline are described. Then, the approaches proposed to tackle the aforementioned limitations are described. Two paths were explored. First approach explores the possibility of preprocessing the input data, namely partially observed scene in a form of a point cloud with additional geometrical information that can be analytically computed offline, to improve the quality of predictions. Second one proposes the architecture *Collision-GraspNet*, which learns to classify grasp proposals on partially observed data as colliding or non-colliding, thus enabling potential filtering or refinement of such proposals.

4.1 Grasp estimation

In the following sections, the two architectures used as a basis for experiments are explained. The first model is used as a baseline model, while the second one is a base model for the proposed extensions. Additionally, the adopted evaluation pipeline is described.

4.1.1 GraspNet

Fang et. al [Fan+20] proposed an end-to-end grasp pose prediction network given point cloud inputs, where the approaching direction and operation parameters are

learned in a decoupled manner. The following subsections discuss the used grasp representation and three main structural elements of *GraspNet*, namely Approach, Operation and Tolerance Networks.

4.1.1.1 Grasp representation

The gripper frame of *GraspNet* is shown in Figure 4.1. Grasp pose detection aims to predict the orientation and translation of the gripper under the camera frame, as well as the width of the gripper. *GraspNet* represents the gripper pose **G** as:

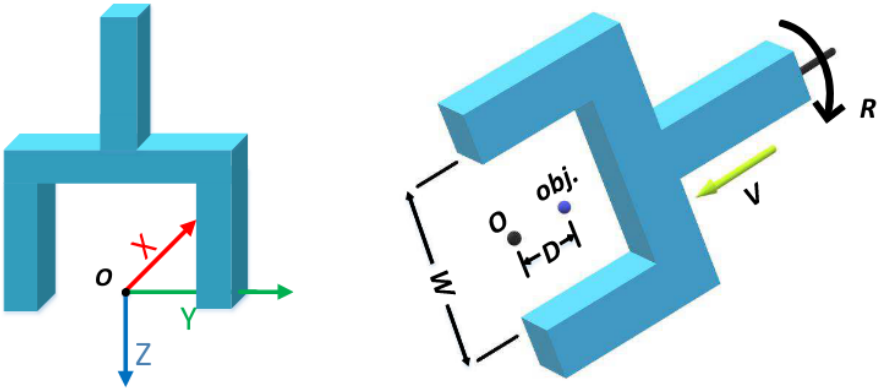


Figure 4.1: Grasp representation in *GraspNet*. (left panel) Coordinate frame of the gripper. (right panel) Pose reformulation: approaching vector V, approaching distance D, in-plane rotation R, gripper width W. Taken from [Fan+20].

$$\mathbf{G} = [\mathbf{R}\,\mathbf{t}\,w], \tag{4.1}$$

where $\mathbf{R} \in \mathbb{R}^{3\times3}$ is the gripper orientation, and $\mathbf{t} \in \mathbb{R}^{3\times1}$ is the center of grasp, while $w \in \mathbb{R}$ is the gripper width. The grasp estimation problem is further reformulated as follows: for the grasp points on the surface of target objects, predict the approaching vectors, approaching distances and in-plane rotations along with the gripper width.

4.1.1.2 Architecture

An overview of the *GraspNet* composition is shown in Figure 4.2.

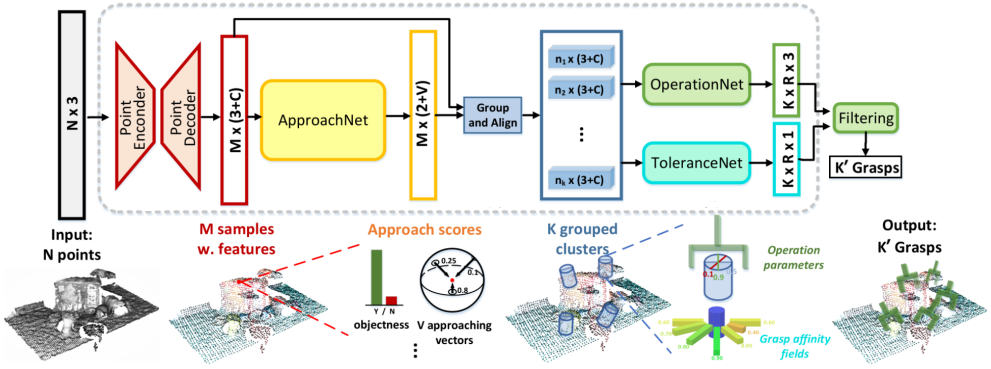


Figure 4.2: Overview of *GraspNet* network. Point encoder-decoder extracts the features of an input point cloud $N \times 3$, samples M points with C features channels. ApproachNet predicts the approaching vectors that are used to group and align the points in cylinder regions. OperationNet predicts the operation parameters while ToleranceNet predicts grasp robustness. Taken from [Fan+20].

ApproachNet (shown in yellow) estimates grasp points jointly since some directions are not feasible for grasping due to occlusion. PointNet2 [Qi+17a] is used as a backbone. Given an input point cloud of size $N \times 3$, a new set of points with C feature channels is produced by the base network (shown in red). M points are subsampled using farthest point sampling to cover the scene in full. The backbone is composed out of four set abstraction and two feature propagation layers.

The shape of the output of the ApproachNet itself is $M \times (2 + V)$: feasible approaching vectors are classified into V predefined viewpoints, and for each point confidence of whether this points is graspable or not is predicted.

Each candidate point gets an assigned binary label that indicates graspability. Points that do not belong to objects have negative labels, and points that do belong to objects are filtered to obtain those that have at least one graspable ground-truth in the 5 mm radius neighborhood. Such points get a 1 graspable label, while the rest is ignored and do not contribute during training.

Each resulting graspable point is subject to approaching vector sampling in the camera frame around the said point. In the following, v_{ij} denotes the vector from j^{th} view of i^{th} graspable point. Reference ground truth vector $\hat{v}_{ij}$ is searched for

in the sphere space around i^{th} point. Additionally, only references that lie withing 5 degree bound are considered. Target function is defined as follows:

$$L^A(\{c_i\},\{s_{ij}\}) = \frac{1}{N_{cls}}\sum_i L_{cls}(c_i, c_i^*) + \lambda_1 \frac{1}{N_{reg}}\sum_i \sum_j c_i^* \mathbf{1}(|v_{ij}, v_{ij}^*| < 5^\circ) L_{reg}(s_{ij}, s_{ij}^*), \tag{4.2}$$

where c_i is the binary prediction of the graspability of the point i, c_i^* is 1 if the point is graspable and 0 otherwise. s_{ij} is the predicted confidence score for j^{th} view of the point i, while s_{ij} is the maximum grasp ground truth confidence for the current view. Indicator function $\mathbf{1}(.)$ constraints the loss on approaching vectors that has a reference in the 5 degree bound, while $|v_{ij}, v_{ij}^*|$ defines the mentioned degree difference. L_{cls} is the two-class softmax loss, and L_{reg} is the smooth L_1 loss.

OperationNet predicts operational parameters such as in-plane rotation, approaching distance and gripper width using the approaching vectors.

Prior to feeding the predicted vectors to the OperationNet, a representation of each proposed grasp candidate is created. Approaching distances are divided into K bins, and for each distance d_k the points from the inside of the cylinder centered at the approaching vector are sampled. Each sampled point is brought into a coordinate system with the origin lying at the grasp point and Z-axis being the approaching vector $\mathbf{v_{ij}}$. $\mathbf{O_{ij}}$ is thus the transformation matrix that is defined as follows:

$$\mathbf{O_{ij}} = [\mathbf{o_{ij}^1}, [0, -\mathbf{v_{ij}^{(3)}}, \mathbf{v_{ij}^{(2)}}]^T, \mathbf{v_{ij}}],\ \mathbf{o_{ij}^1} = [0, -\mathbf{v_{ij}^{(3)}}, \mathbf{v_{ij}^{(2)}}]^T \times \mathbf{v_{ij}}, \tag{4.3}$$

where $\mathbf{v_{ij}^{(k)}}$ is the k^{th} element of $\mathbf{v_{ij}}$. This transformation produces a unified representation and coordinate system for all candidates.

In-plane rotation prediction is handled as a classification task. Given an aligned point cloud as an input, the predictions are the classification scores and normalized residuals for each binned rotation, predicted grasp width and grasp confidence. Due to the symmetrical structure of the gripper, predicted rotations lie in the $[0, 180]$ degrees range. Following is the objective function of the OperationNet:

$$
\begin{aligned}
L^R(R_{ij}, S_{ij}, W_{ij}) = \sum_{d=1}^{K} (& \frac{1}{N_{cls}} \sum_{ij} L^d_{cls}(R_{ij}, R^*_{ij}) \\
& + \lambda_2 \frac{1}{N_{reg}} \sum_{ij} L^d_{reg}(S_{ij}, S^*_{ij}) \\
& + \lambda_3 \frac{1}{N_{reg}} \sum_{ij} L^d_{reg}(W_{ij}, W^*_{ij})),
\end{aligned}
\tag{4.4}
$$

where R_ij is the binned rotation in degrees, S_ij is the grasp confidences, W_ij is the gripper width, and d is the approaching distance. L^d is d^{th} binned distance loss, while L_{cls} is the sigmoid cross entropy loss function, and L_{reg} is the smooth L_1 loss.

ToleranceNet employs grasp affinity fields representation to learn and predict the tolerance to perturbations of each grasp.

For each ground truth grasp pose, its neighbors in the sphere space are searched to estimate the farthest distance that still allows a grasp to be robust with a score > 0.5. Following is the objective function of the ToleranceNet:

$$
L^F(A_{ij}) = \frac{1}{N_{reg}} \sum_{d=1}^{K} \sum_{ij} L^d_{reg}(T_{ij}, T^*_{ij}) \tag{4.5}
$$

where T_{ij} is the maximum perturbation of the grasp pose.

4.1.1.3 Training and inference

During training all modules are trained in an end-to-end fashion with the following objective function minimized:

$$
L = L^A(\{c_i\}, \{s_{ij}\}) + \alpha L^R(R_{ij}, S_{ij}, W_{ij}) + \beta L^F(T_{ij}) \tag{4.6}
$$

During inference, grasp poses are separated into 10 bins in accordance to their grasp scores. Grasps in each bin are sorted in accordance to the maximum predicted perturbation. Additionally, all predictions can be optionally filtered for collision. This is achieved by means of the Model-Free Collision Detection: each measured scene point cloud is downsampled and voxelized with the voxel size of $0.01m$, the corresponding predicted grasp poses are checked for collision with the scene, and if global intersection over union (IoU) of the gripper model and the voxelized scene

exceeds $1cm^3$, then the corresponding prediction is labeled as colliding with the visible scene point cloud.

4.1.1.4 Implementation

GraspNet was implemented using Python 3.6 and PyTorch 1.2. Rotation angles and approaching distances were divided into 12 and 4 bins respectively. Approaching distances bins are 0.01, 0.02, 0.03, and 0.04 meters. The amount of sampled points M was set to 1024, the amount of viewpoints V was set to 300. PointNet2's set abstraction layers employ radii of 0.04, 0.1, 0.2, and 0.3 meters and grouping sizes of 64, 32, 16, and 16 where the point set is downsampled to 2048, 1024, 512, and 256 points. Feature propagation layers upsample to 1024 and 256 channel features. ApproachNet, OperationNet, and ToleranceNet are MLPs with the following sizes:

- ApproachNet: $256, 302, 302$
- OperationNet: $128, 128, 36$
- ToleranceNet: $128, 64, 12$

The loss function 4.6 has the following default parameters:

- $\lambda_1 = 0.5$
- $\lambda_2 = 1.0$
- $\lambda_3 = 0.2$
- $\alpha = 0.5$
- $\beta = 0.1$

The optimizer of choice is Adam, and the number of points N sampled to represent the scene is 20000. The initial learning rate is 0.001 and it is decreased one order of magnitude after 60 and 100 epochs. The batch size is 4.

The introduction of an explicit notion of a colliding proposal grasp pose into a training pipeline may improve the qualitative results of these proposals. First, *Contact-GraspNet* [MS21] is described as a model that will help highlight the effect of not having the aforementioned notion of collision, as well as serve as a basis for proposed *Collision-GraspNet* architecture, which is presented in the second half of this section.

4.1.2 Contact-GraspNet

4.1.2.1 Grasp representation

Contact-GraspNet grasp representation is shown in Figure 4.3.

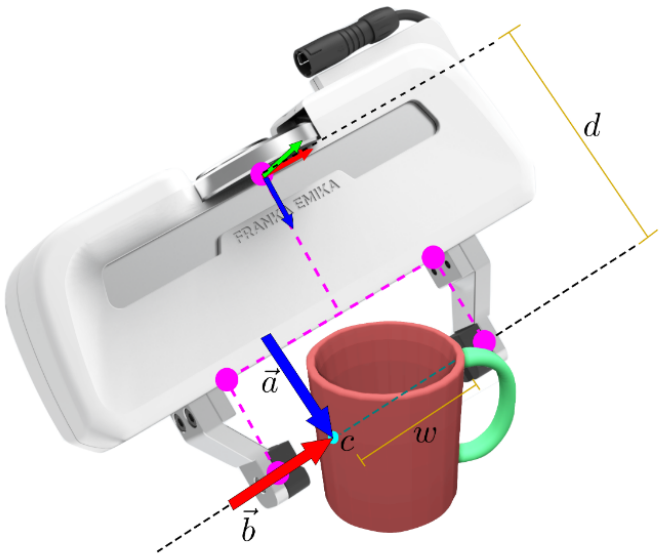


Figure 4.3: Grasp representation in *Contact-GraspNet*. c is an contact point being observed, **a** and **b** are the 3-DoF rotation, w is the predicted gripper finger opening width, while d is the base to baseline distance. Magenta colored points correspond to gripper points **v** that are taken into account in l_{add-s} loss. Taken from [MS21].

In this representation, the fact that for most two-finger grasps at least one of the two gripper contacts is visible prior to grasping is taken into an account. Thus, the distribution of ground truth grasps $g \in G$ is mapped to the corresponding contact point $c \in \mathbb{R}^3$. Additionally, their location in 3D is represented by the neighboring points in the point cloud.

Without the loss of generality, the problem of 6-DoF estimation is reduced to 3-DoF grasp rotation $R_g \in \mathbb{R}^{3\times3}$ and grasp width $w \in \mathbb{R}$ for a two-finger parallel-jaw gripper. The contact point $\mathbf{c} \in \mathbb{R}^3$ of a gripper's pad center is the base point for defining the whole 6-DoF grasp pose $g \in G$ representation $(R_g, t_g) \in SE(3)$ and

grasp width $w \in \mathbb{R}$:

$$\begin{aligned} \mathbf{t_g} &= \mathbf{c} + \frac{w}{2}\mathbf{b} + d\mathbf{a} \\ R_g &= [\mathbf{b} \quad \mathbf{a} \times \mathbf{b} \quad \mathbf{a}], \end{aligned} \tag{4.7}$$

where $\mathbf{t_g}$ and R_g are the translation and rotation components of pose g, respectively.

4.1.2.2 Architecture

Contact-GraspNet employs PointNet2 [Qi+17a] feature abstraction and feature propagation layers to build an asymmetric U-shaped network which allows for efficient processing of point clouds and hierarchical aggregation of their feature representations. An overview of the whole training pipeline is presented in Figure 4.4.

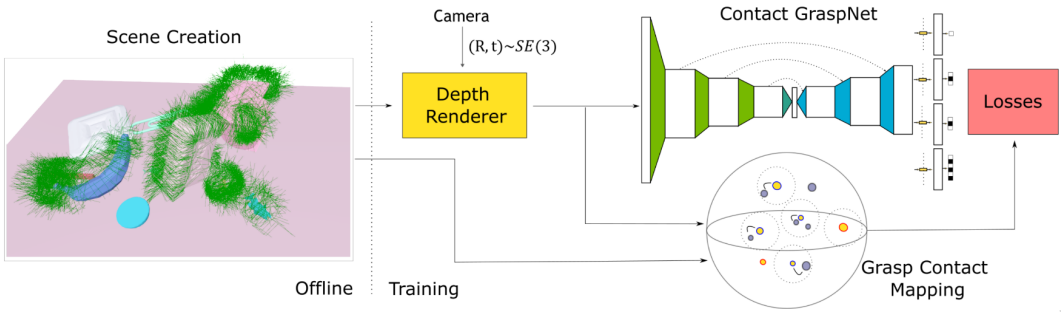


Figure 4.4: Overview of *Contact-GraspNet* pipeline. Offline scene composition using *ACRONYM* dataset or its derivatives is followed by online valid grasp mapping to their contact points on the mesh surface. Virtual camera views are sampled in the scene to render point depth maps. Recorded points (yellow) are annotated as positive if there is a contact on the mesh (blue) in a 5 mm radius, and are associated with the corresponding transformation of the said mesh contact. Resulting point annotations are used for supervision. Taken from [MS21].

Input data is represented by a set of N random points $p \in \mathbb{R}^{N\times 3}$ sampled from the recorded input point cloud. The resulting prediction is made on M farthest points. Each of the four output heads has two 1D convolutional layers and provides the following per-point outputs that are used to form a grasp representation.

$\mathbf{o} \in \mathbb{R}^{10}$ is used to obtain grasp width $\hat{\mathbf{w}}_\mathbf{i} \in [0, w_{max}]$ that is split into 10 bins $\hat{\mathbf{o}} \in \mathbb{R}^{10}$, and the center value with the highest confidence represents $\hat{\mathbf{w}}_\mathbf{i}$.

$\mathbf{z_1} \in \mathbb{R}^3$ and $\mathbf{z_2} \in \mathbb{R}^3$ are used to obtain the approach direction $\mathbf{a} \in \mathbb{R}^3$ and the baseline direction $\mathbf{b} \in \mathbb{R}^3$. Since they are orthonormal due to the grasp representation definition, both direction predictions $\hat{\mathbf{a}}, \hat{\mathbf{b}}$ are coupled in an in-network Gram Schmidt orthonormalization procedure to decrease the dimensionality of grasp predictions by only predicting $\hat{\mathbf{a}}$ as the orthonormal component of $\hat{\mathbf{b}}$:

$$\hat{\mathbf{b}} = \frac{\mathbf{z_1}}{||\mathbf{z_1}||} \tag{4.8}$$

$$\hat{\mathbf{a}} = \frac{\mathbf{z_2} - \langle \hat{\mathbf{b}}, \mathbf{z_2} \rangle \hat{\mathbf{b}}}{||\mathbf{z_2}||}. \tag{4.9}$$

$\hat{s} \in \mathbb{R}$ are the contact success predictions. During training for each rendered point cloud $\mathbf{P} = \mathbf{p_1}, ..., \mathbf{p_n} \subset \mathbb{R}^3$ a point-wise grasp success is assigned in the following manner:

$$\forall i = 1, ..., n \; s_i = \begin{cases} 1 & min_j ||\mathbf{p_i} - \mathbf{c_j}||_2 < r, \\ 0 & \text{otherwise}, \end{cases} \tag{4.10}$$

where $\mathbf{c_j} \in \mathbf{P}$ are the mesh contact points of non-colliding dataset ground truth grasps $\mathbf{g_j} \in G$ in camera coordinates, and $r \in \mathbb{R}$ is the maximum contact mapping radius. Given $\mathbf{P}^+ \subset \mathbf{P}$ that have feasible grasp contacts within radius r, for each point $\mathbf{p_i^+} \in \mathbf{P}^+$ the closes grasps are assigned in the following way:

$$\begin{bmatrix} w_{g,i} \\ R_{g,i} \\ \mathbf{t}_{g,i} \end{bmatrix} = \begin{bmatrix} w_{g,j} \\ R_{g,j} \\ \mathbf{p}_i^+ + \frac{w_j}{2} \mathbf{b}_j + d\mathbf{a}_j \end{bmatrix}, \tag{4.11}$$

with

$$j = argmin_k ||\mathbf{p}_i^+ - \mathbf{c}_k||_2. \tag{4.12}$$

$\hat{s} \in \mathbb{R}$ are evaluated at all output points $\mathbf{p}_i \in \mathbb{R}^3 : \forall i \in [0, m]$ using binary cross entropy $l_{bce,k}$. Top k predictions are propagated, other predictions concerning the geometry of grasps are evaluated only at $\mathbf{p}_i^+$. During training all output heads are combining their predictions to form a 6-DoF grasp pose $\hat{g} \in G$ following 4.7. Five points $\mathbf{v} \in \mathbb{R}^{3 \times 5}$ that are shown in Figure 4.3 are transformed into ground truth

frames computed following 4.11. $\mathbf{v}$ is transformed into the predicted grasp poses at $\mathbf{p}_i^+$:

$$\begin{aligned} \mathbf{v}_i^{gt} &= \mathbf{v}R_{g,i}^T + \mathbf{t}_{g,i}, \\ \hat{\mathbf{v}}_i^{gt} &= \mathbf{v}\hat{R}_{g,i}^T + \hat{\mathbf{t}}_{g,i}. \end{aligned} \tag{4.13}$$

6-DoF grasp loss l_{add-s} is defined as a weighted sum of average distances between the gripper ground truth points $\mathbf{v}$ and $\hat{\mathbf{v}}$.

$$l_{add-s} = \frac{1}{n^+} \sum_{i}^{n^+} \hat{s}_i min_u ||\hat{\mathbf{v}}_i - \mathbf{v}_u^{gt}||_2, \tag{4.14}$$

where n^+ is a size of P^+ and $\hat{s}_i$ is the predicted contact success confidence. For the predicted binned grasp widths the weighted multi-label binary cross entropy l_{width} is optimized.

4.1.2.3 Training and inference

During training the following target function is minimized:

$$L = \alpha l_{bce,k} + \beta l_{add-s} + \gamma l_{width} \tag{4.15}$$

For inference, the point cloud is centered at its mean in camera coordinates. Additionally, during inference the local regions of interest can be extracted as centroids of the input point cloud segments in order to increase the number of potential valid contact points. For this purpose, cubes with an edge size set to doubled largest spanning dimension, but no less than 0.3 m and no more than 0.6 m are extracted.

During inference, the predicted grasp poses are selected according to their confidence values. First, the grasps with $\hat{s} > th_{s,1}$ are selected, and then farthest point sampling is used to ensure broad grasp coverage. If the number of the selected grasps is less than 2048, the grasps with $\hat{s} > th_{s,2}$ are also selected.

4.1.2.4 Implementation

Contact-GraspNet was implemented using Python 3.7 and TensorFlow 2.1. The number of points of the point cloud N was set to 20000, and M was set to 2048.

The three set abstraction layers have the following query ball radii: $[0.02, 0.04, 0.08]$ for the first, $[0.04, 0.08.0.16]$ for the second, and $[0.08, 0.16, 0.32]$ for the third, respectively. Output sizes for the MLPs of feature propagation layers are $(256, 256)$, $(256, 128)$, and $(128, 128, 128)$, respectively.

The loss function 4.15 has the following default parameters:

- $\alpha = 1$,
- $\beta = 10$,
- $\gamma = 1$,

Contact mapping radius r is set to $0.005m$. The optimizer of choice is Adam with an initial learning rate of 0.001, decay rate of 0.7 for each 200000 samples. The confidence thresholds $th_{s,1}$ and $th_{s,2}$ are set to 0.23 and 0.19, respectively. Batch size is 3 for 144.000 iterations.

4.1.2.5 ACRONYM-FORK01

The *ACRONYM-FORK01* dataset provides training data for the *Contact-GraspNet*. *ACRONYM-FORK01* is a direct derivative of *ACRONYM* dataset [EMF20], see Chapter 3.3 for more details on *ACRONYM*.

ACRONYM-FORK01 uses meshes provided in *ACRONYM* to generate 10000 scenes, where each scene is composed of $8 - 12$ randomly sampled object meshes placed on a support object in stable poses. Each scene is represented by an *.npz* file containing source mesh paths and the corresponding object poses. Additionally, a varying number of ground truth non-colliding grasps is stored for each scene, where each grasp is represented by the corresponding contact points $c_i \in \mathbb{R}^{2\times3}$ and pose $g \in R^{4\times4}$. The generated contact points are computed on scene meshes using the simplified Franka Panda gripper model with a 0.1 m finger opening. Visual comparison of the original gripper model and the used simplified one can be seen in Figure 4.5.

4.1.3 Evaluation

Fang et. al. [Fan+20] proposed a unified online 6-DoF evaluation pipeline. Each predicted grasp pose $\mathbf{P_i}$ is associated the target object by checking the pointcloud

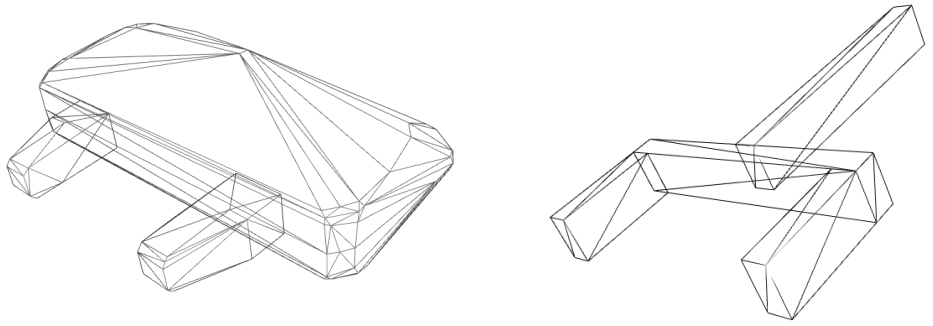


Figure 4.5: Gripper models: original Franka Panda gripper (left panel), simplified version with 0.1 m finger opening (right panel).

inside the gripper opening. Then the force-closure metric is employed to provide a binary label for each predicted pose given different values of friction coefficient μ in a similar way that was used during grasp annotation generation for the *GraspNet-1Billion* dataset.

Since it is expected that for a cluttered scene setup there are multiple grasp poses to be predicted, the percentage of true positive is more important. Thus the $Precision@k$ metric is adopted. This metric outputs the precision of top k ranked grasps. For each friction coefficient μ value $k = 50$ top grasps are evaluated, defining an average precision AP_μ value. In the scope of this work AP value for each dataset test split is reported: average precision over all μ values in the range $[0.2, 1.2]$ with $\Delta\mu = 0.2$ as interval. Additionally, each prediction is tested for collision with a fully visible scene. For this purpose target scenes is downsampled and voxelized with a voxel edge size of $0.008m$, and predictions are checked for intersection with the occupied voxel space.

In order to not allow similar grasp poses or grasp poses for single object to dominate the scene, grasp pose non-maximum suppression (NMS) is used before evaluation.

For two grasps $\mathbf{G}_1$ and $\mathbf{G}_2$ a grasp pose distance $D(\mathbf{G}_1, \mathbf{G}_2)$ is defined as

following:

$$D(\mathbf{G}_1, \mathbf{G}_2) = (d_t(\mathbf{G}_1, \mathbf{G}_2), d_\alpha(\mathbf{G}_1, \mathbf{G}_2)) \tag{4.16}$$

where $d_t(\mathbf{G}_1, \mathbf{G}_2)$ is a translation distance and $d_\alpha(\mathbf{G}_1, \mathbf{G}_2)$ is a rotation distance between $\mathbf{G}_1$ and $\mathbf{G}_2$.

Assuming a grasp pose is defined by a translation vector $\mathbf{t}$ and rotation matrix $\mathbf{R}$, then aforementioned distances are defined as following:

$$d_t(\mathbf{G}_1, \mathbf{G}_2) = ||\mathbf{t}_1 - \mathbf{t}_2|| \tag{4.17}$$

$$d_\alpha(\mathbf{G}_1, \mathbf{G}_2) = \arccos \frac{1}{2}(tr(\mathbf{R}_1 \cdot \mathbf{R}_2^T) - 1) \tag{4.18}$$

Since rotation and translation are not lying in the same metric space, NMS threshold TH is defined as a tuple just like pose distance D:

$$TH = (th_d, th_\alpha) \tag{4.19}$$

Then $D(\mathbf{G}_1, \mathbf{G}_2) < TH$ if:

$$d_t(\mathbf{G}_1, \mathbf{G}_2) < th_d, d_\alpha(\mathbf{G}_1, \mathbf{G}_2) < th_\alpha \tag{4.20}$$

Thus, two grasps $\mathbf{G}_1$ and $\mathbf{G}_2$ are merged into one if $D(\mathbf{G}_1, \mathbf{G}_2) < TH$. Only top K grasps per each object in the scene contribute towards the overall score according to confidence scores, while the rest is ignored. $th_d = 3cm$, $th_\alpha = 30°$, and $K = 10$ are used. Result of applying pose-NMS using ground truth grasps from *GraspNet-1Billion* dataset are shown in Figure 4.6.

4.2 Curvature measure

Given the constraint of learning from partially observable scenes, namely being unable to learn from the occluded parts of the scene, certain geometrical properties of 3D data can be exploited to give explicit indication about the geometry of the occluded parts of shapes in the scene. Embedding input data with that information may improve quantitative and qualitative results. The most obvious choice for exploring this hypothesis is using curvature information of a given shape. For curvature estimation on polyhedral approximation of smooth surfaces the modified algorithm based on mean curvature measure is used [tC06].

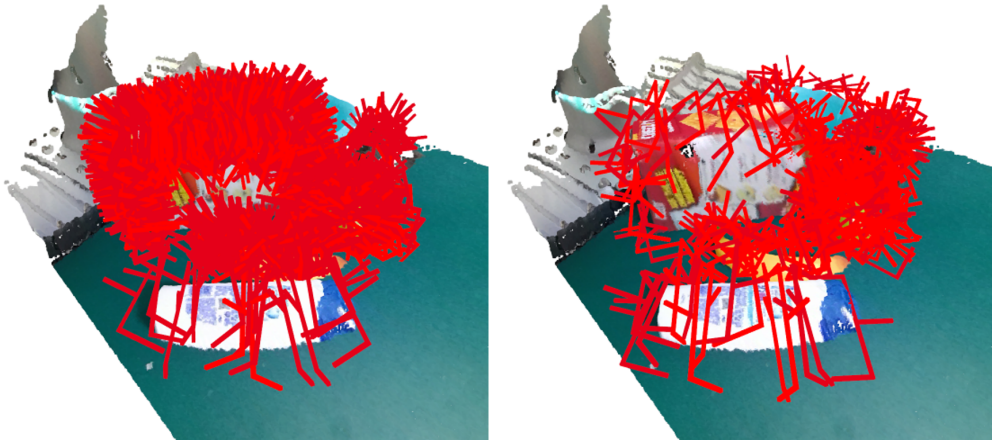


Figure 4.6: Result of applying pose-NMS on dense group of ground truth grasps from *GraspNet-1Billion* dataset: original proposals (left), resulting proposals (right).

First, let M define a surface in oriented euclidean space $\mathbb{R}^3$. M is also assumed to be the the boundary of some compact set $V \subset \mathbb{R}^3$. Assuming M is smooth, the unit normal vector at a point $p \in M$ pointing outward V is noted as $n(p)$. Given a vector v in the tangent plane T_pM to M at p, the derivative of $n(p)$ in the direction v is orthogonal to $n(p)$ as $n(q)$ has unit length for any $q \in M$. The derivative D_pn of n at p is defining an endomorphism of T_pM. Eigenvectors and eigenvalues of T_pM are respectively called *principal directions* and *principal curvatures*. From the trace and determinant of D_pn principal curvatures also called *mean and Gaussian curvature at* p can be recovered.

In smooth case, the mean curvature measure ϕ_V^H is a following function that associates with every set $B \subset \mathbb{R}^3$ the quantity:

$$\phi_V^H(B) = \int_{B \cap M} H(p)\, dp \tag{4.21}$$

where $H(p)$ being the mean curvature of M at point p.

This definition can be adopted for triangulated surfaces, where V is a polyhedron with vertex set P and edge set E. Discrete mean curvature measure ϕ_V^H then is:

$$\phi_V^H(B) = \sum_{e \in E} length(e \cap B)\beta(e) \tag{4.22}$$

where $|\beta(e)|$ is the angle between the normals to the triangles of M incident on e. The sign of $\beta(e)$ is positive if e is convex and negative if it is concave.

In the base case for each vertex p, a query ball $B_r(p)$ with radius r and center at p is defined. The intersection of a candidate edge e with $B_r(p)$ is then a line-sphere intersection segment. The drawback of such methodology is that in case of meshes that have a thin structural segment as its part (mesh of a painting, for example) means that edges spanned by the vertex from 'opposite' side of the mesh will be taken into consideration. This undesirable effect can be mitigated by reducing radius r. However, in practice reducing the radius is often not a viable option since if $B_r(p)$ is too small, the curvature of neighborhood will not be captured properly. Additionally, some meshes of comparatively poor quality, especially in such "thin plane" cases, may project artifacts on the opposing plane, this effect is shown in Figure 4.7.

To properly mitigate this effect, each candidate edge $e \in E$ is checked to be connecting the target vertex p to one of its neighbors, i.e. $e = (u, p) \in E$ where u is neighbor of p. This way, for each vertex point p defined by its coordinate in $\mathbb{R}^3$ of mesh V there is an associated mean curvature measure value.

Prior to embedding the resulting value for each point, the fact that this value is not absolute and not normalized must be taken into account. In order to represent the degree of concavity or convexity of neighborhood around p, positive and negative values are normalized separately:

$$h(p)_+ = \frac{h(p)_+ - b_{+,lower}}{b_{+,upper} - b_{+,lower}} \tag{4.23}$$

$$h(p)_- = 1 - \frac{h(p)_- - b_{-,lower}}{b_{-,upper} - b_{-,lower}} \tag{4.24}$$

where b_+ and b_- are normalization bounds for positive and negative values correspondingly. For positive curvature values it is set to $[0, b]$, while for negative values to $[-b, 0]$, where b is defined empirically. Prior to normalizing, the values are clipped to their corresponding b values. This leads to normalization into $[0, 1]$ range, where the closer the value to 1, the higher degree of convexity/concavity

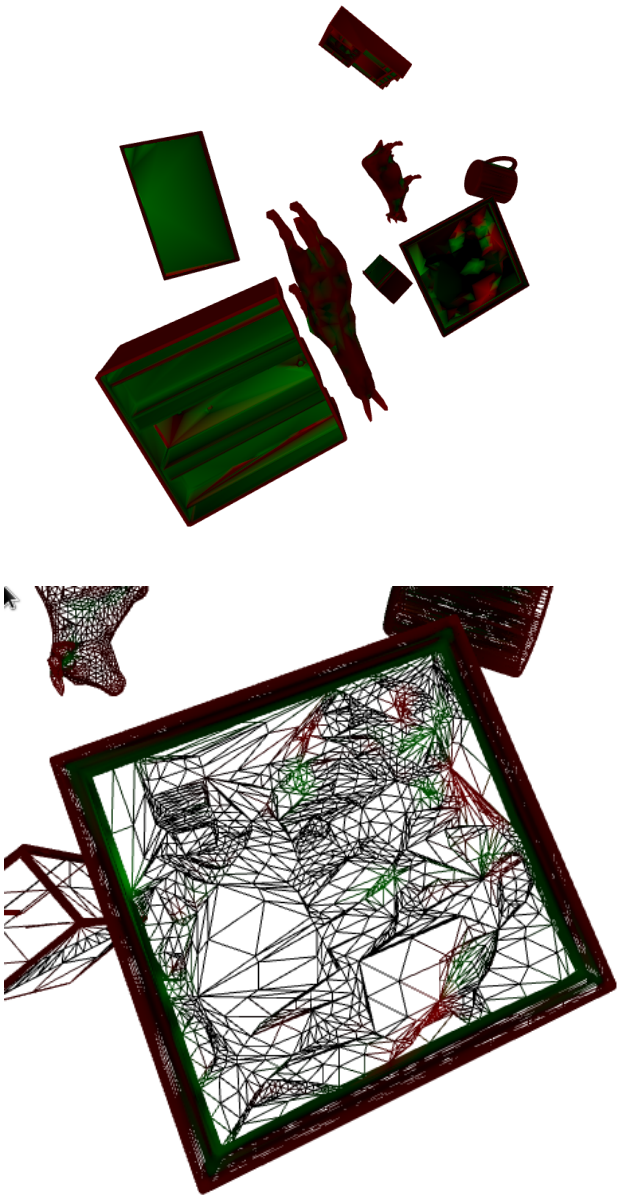


Figure 4.7: Curvature embedding artifacts on the painting mesh (upper panel) on the right side of the image caused by poor quality of the mesh, and its wireframe (lower panel).

it represents. Each normalized set of values is treated as a value of one of RGB channels of vertex colors assigned per each vertex of a mesh. Red channel is

occupied by the normalized values that represent the degree of concavity, and green channel - by values that represent convexity.

4.2.1 ACRONYM-FORK01-CRV

Using the annotations provided in *ACRONYM-FORK01* dataset, *ACRONYM-FORK01-CRV* representation is generated as follows. Each scene in *ACRONYM-FORK01-CRV* is constructed using the provided annotations; the resulting mesh set is concatenated into a single mesh; and each vertex is assigned an RGB color value representing local curvature following the procedure defined in 4.2. The embedded scene representations are exported in the *.ply* format. When computing the modified mean curvature measure, query ball radius r was set to 0.01, and normalization bounds were set to 0.1 and -0.1 for positive and negative values, respectively. Figure 4.8 provides an example of the processed *ACRONYM-FORK01* scene.

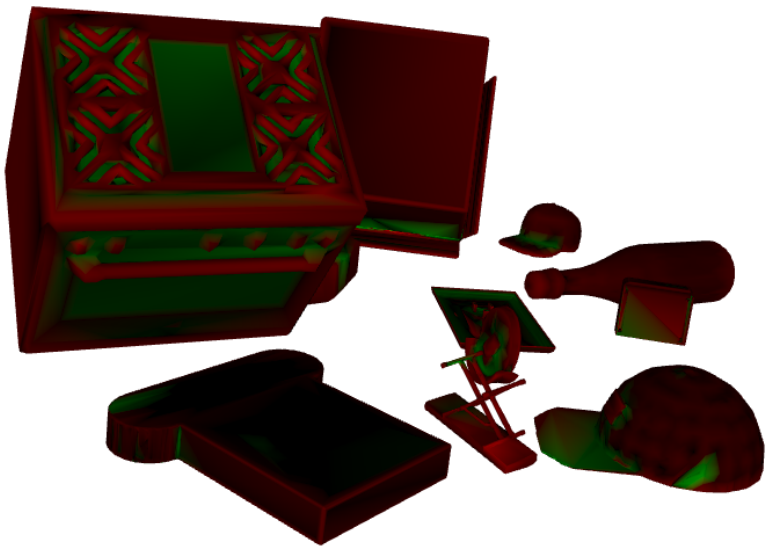


Figure 4.8: Curvature-embedded scene from *ACRONYM-FORK01-CRV* dataset.

4.2.2 GraspNet-CRV

The second derivative dataset used in this work is *GraspNet-CRV* which builds upon the annotations and data provided by the *GraspNet-1Billion* dataset [Fan+20] (see Chapter 3.2 for more details on *GraspNet-1Billion*).

GraspNet-1Billion provides 190 scenes each containing 256 RGB-D image pairs. For each pair, the object poses as well as the camera pose are annotated. Thus, the aim is to generate a third image for each pair: an RGB image of the curvature-embedded scene. In order to do so, the scenes must be reconstructed using the provided annotations. *GraspNet-1Billion* provides 88 source mesh objects that are used for composing the scenes. These meshes have on average around 500000 vertices each, which means that, on one hand, they are a perfect target for accurate curvature computation, and, on the other, such computation in a scene of 10 objects in general will take a considerable amount of computational effort. For this reason, all meshes are processed prior to the scene construction to reduce the vertex count while preserving the original topology. For this purpose, the quadratic error metric mesh simplification algorithm [GH97] is employed.

The quadratic error metric mesh simplification algorithm is based on the iterative contraction of vertex pairs. To be more specific, starting with the initial model M, a sequence of pair contractions is applied until the simplification goals are satisfied and the final approximation M^* is produced. The simplification goal in this case is the desired amount of vertices m_p. In order to perform vertex pair contraction and choose new vertex location, the cost of contraction is defined. For vertex $p = [x, y, z, 1]^T$ and plane $q = [a, b, c, d]^T$, the squared distance $dist(q, p)^2$ between q and p is:

$$dist(q,p)^2 = (q^T p)^2 = p^T(qq^T)p = p^T Q_p p \tag{4.25}$$

where Q_p is the following 4×4 matrix:

$$Q_p = qq^T = \begin{bmatrix} a^2 & ab & ac & ad \\ ab & b^2 & bc & bd \\ ac & bc & c^2 & cd \\ ad & bd & cd & d^2 \end{bmatrix} \tag{4.26}$$

This way, $p^T Q_p p$ is a quadratic form representing the approximation error at

vertex p. Since each vertex has a set of planes (triangles) associated with it, the approximation error of each vertex is defined as the sum of squared distances to all planes:

$$\Delta(p) = \sum_i p^T Q_{q_i} p = p^T (\sum_i Q_{q_i}) p \tag{4.27}$$

For a given contraction $(p_1, p_2) \to \hat{p}$ a $\Delta(\hat{p}) = \Delta(p_1) + \Delta(p_2)$ matrix is computed which approximates the error at $\hat{p}$. In order to actually perform the contraction, a new vertex position $\hat{p}$ must be computed that minimizes $\Delta\hat{p}$. Since the error function is quadratic, finding its minimum is a linear problem. Thus, $\hat{p}$ is found by solving $\delta\Delta(\hat{p})/\delta x = \delta\Delta(p_1)/\delta y = \delta\Delta(p_2)/\delta z = 0$, which is equivalent to solving:

$$\begin{bmatrix} q_{11} & q_{12} & q_{13} & q_{14} \\ q_{21} & q_{22} & q_{23} & q_{24} \\ q_{31} & q_{32} & q_{33} & q_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} \hat{p} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \tag{4.28}$$

If this matrix is not invertible, the optimal vertex along the line segment (p_1, p_2) is attempted to be found. If it is impossible, the final location is chosen from the endpoints and the midpoint of this line segment.

Such a contraction procedure allows to reduce the time required to process each scene significantly. A visual comparison of an original high-polygon model from the *GraspNet-1Billion* dataset and its decimated version can be seen in Figure 4.9. It can be observed that a significant reduction in the vertex count does not lead to a considerable degradation of object geometry.

For each annotated camera pose a scene was constructed, meshes in the scene were concatenated, and the modified mean curvature measure is computed with the query ball radius r set to 0.01. The normalization bounds were set to 0.1 and -0.1 for positive and negative values, respectively. All source meshes were decimated to approximately 1000 vertices each. An example rendering is shown in Figure 4.10.

4.3 Collision-GraspNet

In the case of grasp pose estimation in partially observable scenes the explicit notion of collision with the occluded geometry of the scene may improve the quality

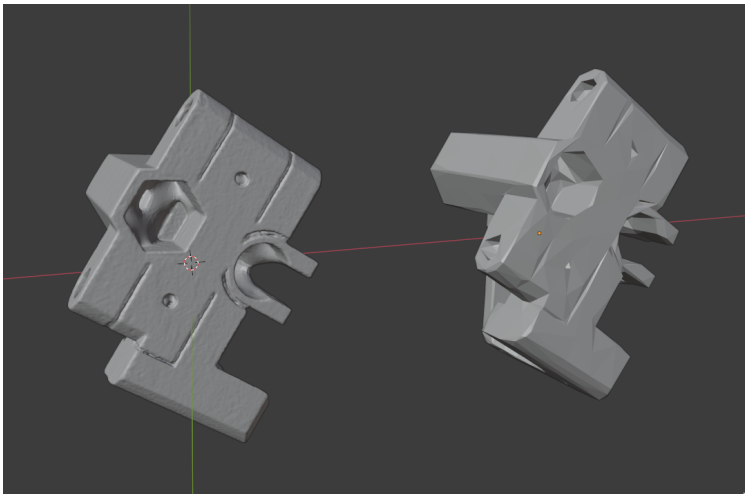


Figure 4.9: Object mesh models: (left) original model from *GraspNet-1Billion* with 748049 vertices, (right) decimated version with 992 vertices.

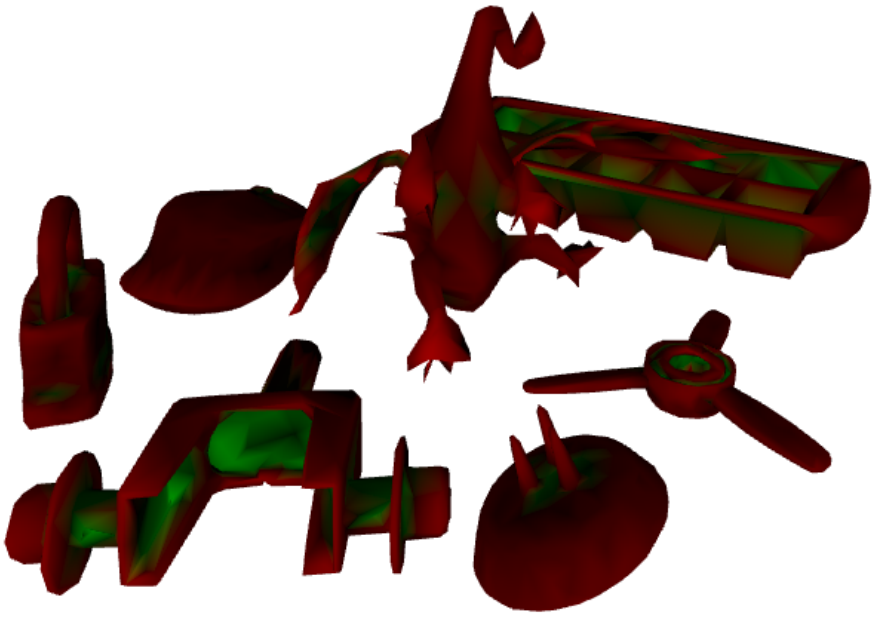


Figure 4.10: Example of a curvature measure embedded *GraspNet-1Billion* scene.

of predictions. *Collision-GraspNet* aims to introduce this notion by assuming a role of the supervisor or refinement network that can evaluate grasp pose proposals on the grounds of collision in order to filter them out or to attempt to refine them.

4.3.1 Architecture

Figure 4.11 provides an overview of the *Collision-GraspNet* training pipeline.

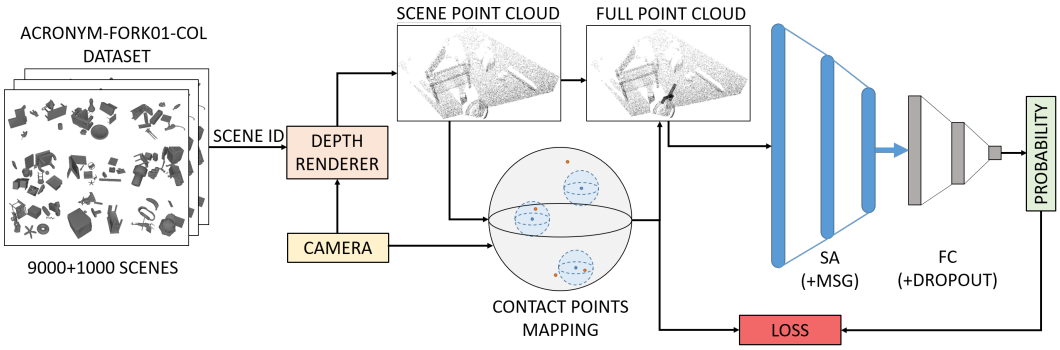


Figure 4.11: A camera pose in $SE(3)$ is sampled (yellow), then a depth map of the target scene is rendered given a randomly sampled scene (orange) from the *ACRONYM-FORK01-COL* dataset with consecutive computation of the scene point cloud of N_{scene} points. Camera pose and scene point cloud are used to map all ground truth contact points of the target scene to the scene point cloud, which allows to draw a ground truth gripper pose sample and use it to obtain a full sample point cloud of $N_{scene} + N_{gripper}$ points which is passed to the network (blue and gray) and corresponding sample class label and outputted class probability (green) are used for loss computation (red).

Collision-GraspNet employs Pointnet2 [Qi+17a] blocks to build a network. The base version of *Collision-GraspNet* uses three set abstraction (SA) layers in total. Such layer takes a point cloud of size $(B, N_{scene} + N_{gripper}, d + C)$, where B is the batch size, N_{scene} is the number of points representing a partially observed scene, $N_{gripper}$ is the amount of points representing the gripper fork, and C is the amount of channels representing auxiliary features. In its turn, an output of such layer is of size $(B, N', d + C')$, where N' is the number of subsampled points with d-dim coordinates and new C'-dim feature vectors summarizing local

context. Each abstraction layer includes three following structural parts. The first is the **sampling layer** where a subset of the input points is chosen using an iterative farthest point sampling method. The second is the **grouping layer** where a point set of size $(N, d+C)$ is transformed into a number of point sets of size $(N^{'}, K, d+C)$ where each group correspond to a local region, and K is the number of points in the neighborhood of centroid points. Since the neighborhood of a point is defined by metric distance, ball query finds all points that are within a radius to the query point, thus defining a neighborhood. The last element is the **PointNet layer**, where an input set of size $(N^{'}, K, d+C)$ is transformed into size $(N^{'}, d+C^{'})$, where each local region in the output is abstracted by its centroid and local feature that encodes the centroid's neighborhood. Layers employ Multi-Scale Grouping (MSG) technique to capture multi-scale patterns of an input point set by applying grouping layers with different scales followed by according PointNets to extract features of each scale. SA layers are followed by three fully-connected layers with dropout layers that output a single predicted class probability $\hat{y}$.

In order to compute $\hat{y}$ a rendered scene point cloud $\mathbf{P} = \mathbf{p_1}, ..., \mathbf{p_n} \subset \mathbb{R}^3$ is used to obtain ground truth-wise binary labels v_i^{gt} indicating the visibility of said ground truth grasp pose:

$$\forall i = 1, ..., n \; v_i^{gt} = \begin{cases} 1 & min_i ||\mathbf{p_i} - \mathbf{c_j}||_2 < r, \\ 0 & \text{otherwise}, \end{cases} \tag{4.29}$$

where $\mathbf{c_j} \in \mathbf{P}$ are the mesh contact points of dataset ground truth grasps $\mathbf{g_j} \in G$ in camera coordinates, and $r \in \mathbb{R}$ is the maximum contact mapping radius. The set G^+ of successfully mapped grasps $\mathbf{g}^+$ and their corresponding class labels y^+ can be extracted from the ground truth contact point set G. They are used to produce a full scene sample by using a random random visible grasp pose $\mathbf{g}_k^+ \in G^+$ of the required class $y_k \in y$, which are passed to the first SA layer of *Collision-GraspNet*. An example of complete point cloud sample is shown in Figure 4.12.

4.3.2 Training and inference

During training, binary cross entropy target loss is minimized:

Figure 4.12: Full point cloud sample containing 20000 points of the partially visible scene (black) and 2000 points of the fork gripper in a non-colliding ground truth grasp pose (green).

$$L = -\frac{1}{B}\sum_{k=1} y_k \log(\hat{y}_k) + (1 - y_k)\log(1 - \hat{y}_k) \tag{4.30}$$

where B is the predicted probabilities output size (batch size), $\hat{y}_k$ is the k-th predicted class label, and y_k is the ground truth class label.

During training, the contact points mapping attempts to balance the amount of valid and colliding ground truth grasp pose examples for each scene by means of sampling (with replacement if needed) to reach the required amount of examples G^+_{max}. Additionally, during both training and inference, each batch is attempted to be populated with the examples of both classes in equal frequency. If for some class it is impossible to create a novel sample, an already sampled example is repeated.

If a sample is classified as colliding during inference, an attempt to refine it can be made. In order to do so, the rotation and the translation component of the proposal are sampled to define the new proposal (R_{pert}, t_{pert}). The rotation component R_{pert} is a dot product between an original rotation matrix R and a

perturbation matrix that rotates the space about a random vector through at a random rotation angle sampled uniformly in $[0, R_{max}]$ range. The translation component t_{pert} is a normal translation vector in a random direction scaled by the uniformly sampled value in $[0, t_{max}]$ in meters. For each new perturbed pose (R_{pert}, t_{pert}), the gripper point cloud is being transformed into the same coordinate frame as the scene point cloud, and this new sample is evaluated by the network. If the sample is classified as a non-colliding grasp and there are scene points in the gripper finger opening (which indicates that the gripper would actually grasp an object for this pose), the perturbation is assumed to be successful and the sampled pose is accepted. *Collision-GraspNet* attempts refinement for a fixed amount of tries n_{tries}.

4.3.3 Implementation

Collision-GraspNet was implemented using Python 3.7 and TensorFlow 2.1. Virtual camera poses were sampled in a quarter sphere above the center of the scene support structure. For each scene, $G^{+}_{max} = 1000$ ground truth samples were used for contact points mapping with the label ratio of $1:1$. The contact mapping radius was set to 0.005 m. The number of points of the scene point cloud N_{scene} was set to 20000, and $N_{gripper}$ to 2000. The gripper's full point cloud was sampled using even farthest point sampling. The first SA layer downsamples the scene to 512 points, while the second to 128 points. Set abstraction layers have the following query ball radii: $[0.1, 0.2, 0.4]$ and $[0.2, 0.4, 0.8]$ in meters. In each local region, the layers sample $16, 32, 128$ and $32, 64, 128$ points for the first and the second layer, respectively. Fully-connected layers have output sizes of 512, 256 and 1 for the first, the second, and the third, respectively. The perturbation parameters R_{max} and t_{max} were set to 0.261 radians and 0.02 meters, respectively. Dropout layer's activation probability is 0.4. The maximum amount of refinement tries n_{tries} was set to 100. The optimizer of choice is Adam. The starting learning rate is 0.001 with a fixed decay rate of 0.7 for each 20000 samples using an exponential decay scheduler. The batch size is 12.

4.3.4 ACRONYM-FORK01-COL

Collision-GraspNet requires a custom dataset for training. Such dataset must contain a representative selection of colliding and valid grasp poses for scenes of structured clutter.

Therefore, another derivative of *ACRONYM* dataset was generated, namely *ACRONYM-FORK01-COL*. This dataset contains 10000 scenes, where each scene contains 8 to 12 object in stable poses. Each object has a variable amount of the ground truth grasps assigned. Each ground truth grasp has one of the two possible labels: colliding grasp (1), or valid grasp (0). Each scene has a fixed ground truth label ratio of $1:1$. All valid grasps are the grasps that were valid for an individual object in an isolated setting and that remained collision-free in the cluttered scene. Half of the colliding grasps are the grasps that were valid for an individual object in isolated setting but are now in collision in the cluttered scene. The second half are the grasps that were sampled from the valid grasps pool and were subsequently perturbed in order to simulate grasps that are close to being valid, yet are in collision with the scene. In order to achieve this, each individual sampled valid grasp has its pose perturbed by individual sampling of the new rotation component R_{pert} and of the translation component t_{pert}, where the maximum perturbation angle R_{max} was set to 0.261 radians, and maximum translation distance t_{max} was set to 0.02 meters. Perturbation was attempted for a fixed amount of tries n_{tries} of 20. All the grasps have corresponding poses and contact points stored, as well as assigned class labels.

5 Experiments and results

In the following, the experiments carried out during the work presented herein are described. The experiments were carried out in three series to test the hypotheses proposed in chapter 4. First, the baseline performance scores are established using the methods, datasets and evaluation pipeline described in 4.1. Second, the curvature-based datasets from 4.2 are used to determine whether the curvature information represented in the aforementioned datasets can boost the performance of the *Contact-GraspNet* model. Finally, the *Collision-GraspNet* architecture proposed in 4.3 is tested and evaluated. Each set of experiments includes a discussion of results.

5.1 Establishing a baseline

The first series of experiments was aiming at establishing the baseline performance scores of both *GraspNet* and *Contact-GraspNet* architectures under the requirements and limitations imposed by the evaluation pipeline described in 4.1. In total, the first series includes seven experiments. These experiments define a performance benchmark for *Contact-GraspNet* making it possible to evaluate the impact of the methods proposed herein.

5.1.1 GraspNet

5.1.1.1 Experiments

The codebase of the *GraspNet* model [Fan+20] was not released initially (as well as the code for the evaluation pipeline proposed in the same publication), thus the first experiment was aimed at reproducing the scores presented by Fang *et al.* The default parameters were used for training and inference (see 4.1.1.4).

Approximately 6 months after the release of the GraspNet-1Billion paper, the codebase was released. The released code does not use the parameters presented in the paper and additionally includes a postprocessing inference step that employs the Model-Free Collision Manager module (MFCM) to filter out the colliding grasps. Two experiments were carried out using the official codebase: the first one establishes the baseline performance of the released model without this postprocessing step, and the second one includes it in order to establish its effectiveness. In both experiments, during training a scene was represented by 20000 points, the number of viewpoints was set to 400, training was conducted for 18 epochs with a batch size of 2. The initial learning rate of 0.001 was a subject to a fixed learning rate decay of 0.1 at $8th$, $12th$, and $16th$ epochs.

During the postprocessing step, the collision threshold was set to $0.01m^3$, and voxel size to $0.01m$. For training *GraspNet-1Billion* dataset's first 100 scenes were used, while all predictions were made on the last 90 scenes (see 3.2 for more information on *GraspNet-1Billion* data splits). Evaluation was carried out according to the pipeline described in 4.1.3.

5.1.1.2 Results

Table 5.1 provides the obtained performance scores for the author's implementation of the *GraspNet* [Fan+20], and for the two takes on the official *GraspNet* model implementation (without and with the optional filtering step). The scores are reported for individual test data splits.

Model	Seen	Similar	Novel
GraspNet [Fan+20] (ours)	13.411	10.723	3.250
GraspNet [Fan+20]	24.597	20.214	4.793
GraspNet + MFCM [Fan+20]	**41.956**	**37.293**	**12.148**

Table 5.1: *GraspNet* [Fan+20] implementations baseline performance according to the chosen evaluation pipeline. Scores are reported for each individual test data split in *GraspNet-1Billion* dataset.

Our custom implementation of *GraspNet* failed to reproduce the scores presented in the original publication. In contrast, the official released code was not only able

to reproduce them, but the optional filtering step shows it effectiveness by almost doubling the scores.

5.1.1.3 Discussion

The reason for such results, more specifically, for the success of the postprocessing step may be the following: in this step, the scene point cloud is voxelized with a $0.01m$ voxel size, and during the collision detection step, the collisions are detected on the full voxelized scene with a $0.008m$ voxel size. This way, the absolute majority of the grasps that would have been labeled as colliding during the evaluation of the predictions are already rejected during the inference. This is because the voxel size during inference is bigger than during evaluation. This analytical approach ensures that the grasps that have high confidence but are colliding are not taken into account during evaluation due to their earlier rejection, thus they do not affect the overall performance score. Significant performance boost given by MFCM indicates that *GraspNet* itself fails to reliably output collision-free predictions. This highlights the strategy adopted by the *GraspNet*: outputting as many predictions as possible, and analytically filtering out the most obvious cases of collision on a coarse approximation of the visible scene.

5.1.2 Contact-GraspNet

5.1.2.1 Experiments

The next step was to establish the performance of *Contact-GraspNet* under the limitations imposed by the chosen evaluation pipeline. For all experiments in this section, *ACRONYM-FORK01* dataset was used for training, and the same aforementioned 90 test scenes from *GraspNet-1Billion* were used during inference. In order to bridge the gap between the data from both datasets, camera poses used for rendering depth maps were sampled from a quarter sphere above and at a distance range of $[0.7, 1.3]m$ from the center point of the support structure. During inference, local regions of the input point cloud were extracted. During evaluation, the predictions were transformed from the *Contact-GraspNet* grasp representation frame into the *GraspNet* grasp representation frame. Training was carried out twice. The default parameters of *Contact-GraspNet* were used during

the first run, while the second run included preprocessing of the depth maps and the point clouds. First, random noise with the scale factor of 0.001 and clip value of 0.005 was added to the depth maps, which were subsequently smoothed with a Gaussian kernel of size 3×3. For each trained model, inference is carried out twice: with the default confidence threshold value and with the lowered values of 0.18 and 0.15 for the first and the second threshold respectively.

5.1.2.2 Results

Table 5.2 provides the obtained performance scores of the trained *Contact-GraspNet* models. The scores are reported for individual test data splits.

Model	**Seen***	**Similar***	**Novel***
Contact-GraspNet	9.642	12.238	3.700
Contact-GraspNet + $th_{s,1} = 0.18$, $th_{s,2} = 0.15$	9.859	12.496	3.997
Contact-GraspNet + depth augmentation	10.184	11.840	4.052
Contact-GraspNet + depth augmentation+ $th_{s,1} = 0.18$, $th_{s,2} = 0.15$	**14.260**	**15.417**	**5.270**
GraspNet [Fan+20]	24.597	20.214	4.793
GraspNet + MFCM	**41.956**	**37.293**	**12.148**

Table 5.2: *Contact-GraspNet* [MS21] performance according to the chosen evaluation pipeline. Scores are reported for each individual test data split in *GraspNet-1Billion* dataset (* *GraspNet-1Billion* was not involved in training, thus for *Contact-GraspNet* test data splits are composed of novel objects for all three splits). *GraspNet* + MFCM performance is included for comparison.

It can be observed that *Contact-GraspNet* compares unfavorably to the *GraspNet* in terms of the scores, but the data preprocessing and lower thresholds lead to a $1.5 - 4.5\%$ score improvement when compared to the default version.

Sample visualization of the grasps predicted by *Contact-GraspNet* for the *GraspNet-*

1Billion dataset scene 153 is shown in Figure 5.1, where the color of the gripper forks represents the confidence score assigned by the network: red for high confidence and blue for low confidence values.

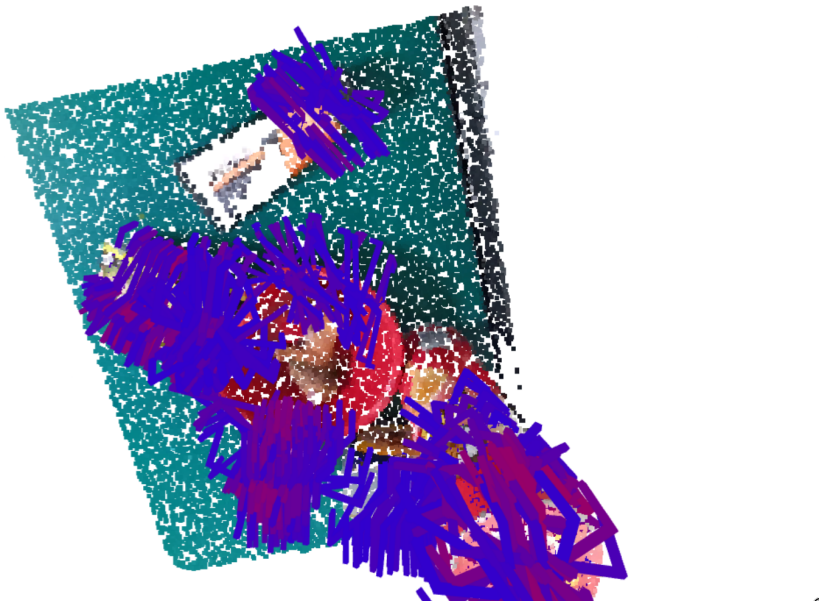


Figure 5.1: *Contact-GraspNet* predictions for the *GraspNet-1Billion* dataset scene 153. Colors of the the gripper forks represents the confidence score assigned by the network: red is for high confidence, blue - for low confidence values.

5.1.2.3 Discussion

There may be multiple reasons for the weaker performance of the *Contact-GraspNet* as compared to the GraspNet. Firstly, there is an unaddressed domain gap between *ACRONYM* dataset synthetic data and *GraspNet-1Billion* dataset real world depth scene measurements. Additionally, all the scenes used for inference are composed of objects that are novel for the trained *Contact-GraspNet* (though the results on the novel test split are considerably lower, which indicates that it is composed of objects of more complex shapes). Then there is a problem of applying grasp NMS during evaluation, which effectively eliminates the dense predictions produced by the *Contact-GraspNet*. And, finally, the evaluation pipeline performs collision detection

on the predictions using the voxelized full scene point cloud computed from the mesh scene representation. This means that some of the colliding predictions can end up in the top-k subset of predictions that defines the overall score for the target scene, thus negatively affecting the performance. Figure 5.2 shows a post-NMS predictions that can be seen in Figure 5.1 that are colored according to their status during evaluation: black color is assigned to the grasps colliding with the voxelized scene representation, gray - to the grasps that are not grasping an object, red and blue - to the valid grasps with high and low network confidence values respectively.

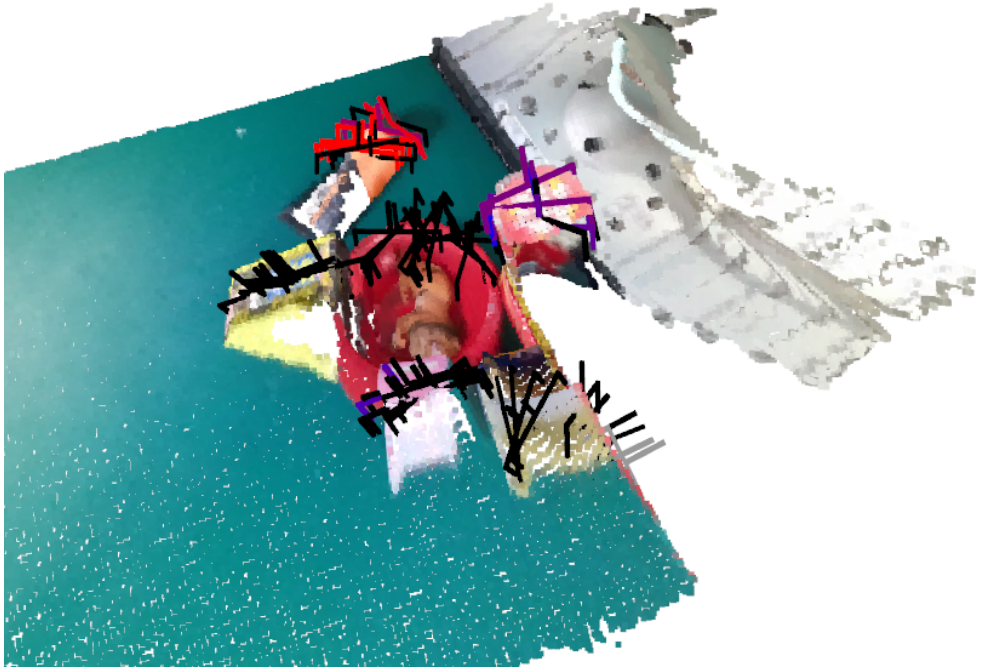


Figure 5.2: *Contact-GraspNet* predictions for the *GraspNet-1Billion* dataset scene 153 after applying grasp-NMS. Colors of the the gripper forks denote their status during evaluation: black color is assigned to the grasps colliding with the voxelized scene representation, gray - to the grasps that are not grasping an object, red and blue - to the valid grasps with high and low network confidence values respectively.

In Table 5.2 one can see that preprocessing and lowering the confidence thresholds

improves the overall scores. The preprocessing step reduces the gap between training and test data, while the reduced confidence thresholds ensure that the network outputs more predictions to consider during the evaluation, reducing the impact of grasp-NMS. However, the reduced thresholds have an undesired implication: the network may be outputting predictions of lesser quality, which means that predictions below a certain threshold may be, for example, colliding with the scene point cloud. For the predictions of the *Contact-GraspNet* with preprocessing and reduced confidence thresholds, 664145 out of 1800424 predictions are in collision with the mesh representation of the scene.

5.2 Curvature measure

The first series of experiments established the performance scores and pinpointed the issues that *Contact-GraspNet* faces under the limitations imposed by the datasets and the evaluation pipeline used. The second series of experiments aims at evaluating the effectiveness of the proposed curvature measure method and the respective generated datasets.

5.2.1 Experiments

In order to enable the *Contact-GraspNet* to learn from the curvature-embedded data that is computed offline, the data pipeline was modified. During training, the sampled camera poses used for rendering the depth maps of the target scenes from *ACRONYM-FORK01* dataset were also used to render an RGB image of the same curvature-embedded scene from *ACRONYM-FORK01-CRV* dataset. This RGB image is passed along with the depth map, regularized to fit the target size of 20000 points per scene. During inference, the pre-rendered RGB images of *GraspNet-CRV* dataset are used as a complimentary part of the depth maps provided by the *GraspNet-1Billion* dataset. The resulting feature vectors are passed to the network's feature abstraction layers in a separate channel during both training and inference.

Similarly to the fist series of experiments, the model was trained twice: as a default *Contact-GraspNet*, and with the optional input data preprocessing step. During inference, the second model used lowered confidence thresholds. The

resulting models are denoted as *Contact-GraspNet-CRV*.

5.2.2 Results

Table 5.3 provides the obtained performance scores for both *Contact-GraspNet-CRV* models trained on the curvature-embedded datasets.

Model	Seen	Similar	Novel
Contact-GraspNet-CRV	8.282	11.127	3.776
Contact-GraspNet-CRV + preprocessing + $th_{s,1} = 0.18$, $th_{s,2} = 0.15$	**8.602**	**11.272**	**3.912**
Contact-GraspNet + preprocessing + $th_{s,1} = 0.18$, $th_{s,2} = 0.15$	**14.260**	**15.417**	**5.270**

Table 5.3: *Contact-GraspNet-CRV* performance according to the chosen evaluation pipeline. Scores are reported for each individual test data split in *GraspNet-1Billion* dataset. Baseline *Contact-GraspNet* performance is included for comparison.

Notably, *Contact-GraspNet* models that used curvature-embedded data failed to produce higher scores than their stock pointcloud-only counterparts. Interestingly, the scores reported in this experiment are lower across all test data splits except for the one that contains the novel objects.

5.2.3 Discussion

It appears that the networks failed to learn useful patterns from the curvature features supplied with the point clouds during training. The reason for such behavior may lie in the quality of the used datasets, namely *ACRONYM-FORK01-CRV* and *GraspNet-CRV*. *ACRONYM-FORK01-CRV* has serious issues with the source objects mesh quality, which results in a poor embedding. One example is presented in Figure 5.3, where a curvature-embedded mesh with a clearly wrong mesh wiring is shown. Such an embedding does not have any value learning-wise, and at best can serve as an added noise.

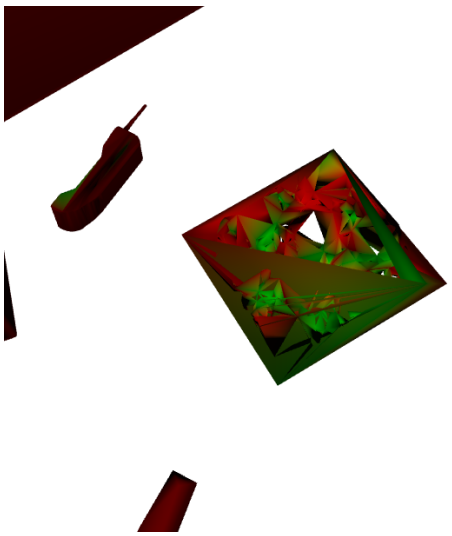


Figure 5.3: Curvature-embedded mesh with poor vertex wiring from *ACRONYM-FORK01-CRV* dataset.

GraspNet-CRV sometimes fails to capture the concavity/convexity properties of objects and produces noise output. An example scene is shown in Figure 5.4.

Figure 5.4: Botched curvature embedding scene from *GraspNet-CRV* dataset.

A common issue for both datasets is the fact that finding the right curvature values normalization bounds that will work across the whole 10000 scenes of

ACRONYM-FORK01 and 23040 annotations of the *GraspNet-1Billion* test data is a non-trivial problem. It is physically impossible to inspect each image and embedding. Additionally, there is a problem of establishing the best decimation factor/target vertex count for the source meshes used in *GraspNet-CRV* dataset. As it was mentioned above, the higher the vertex count, the better is the embedding result, but the slower is the computation. The proposed curvature measure computation method scales poorly with the increasing vertex count in terms of the computation time: the generation of *ACRONYM-FORK01-CRV* took around 160 hours (almost a week), while the generation of *GraspNet-CRV* took around 220 hours (9 days). Overall, according to the obtained scores, both datasets do not have any considerable value for learning-based methods, and more precise tuning of the used parameters is required.

5.3 Collision-GraspNet

The third and final series of experiments is aimed at assessing the effectiveness of the proposed *Collision-GraspNet* architecture using a multistage evaluation process. During the first assessment stage, experiments that determine the optimal parameter and architecture composition were carried out in three sets using the classification accuracy on the *ACRONYM-FORK01-COL* datasets test data split as an effectiveness metric. During the second stage, the best performing model from each set was then evaluated against the analytical model-free collision classification approach employed by *GraspNet*. In this step, the predictions produced by the best performing *Contact-GraspNet* model from the fist series of experiments (see Table 5.2) were used. Finally, in the third stage, the best-performing model is assessed in terms of its capability to improve the said predictions according to the employed pose perturbation methods.

5.3.1 Evaluation on the ACRONYM dataset

The first stage of the assessment consists of a total of ten experiments: one experiment to establish the baseline performance of *Collision-GraspNet*; and three sets of three experiments each that aim to estimate the training parameters and model composition required to produce the best possible classification accuracy.

Notably, the three sets of three experiments were conducted in succession, where the results of one set of experiments would determine which hyperparameter are to be tuned in the next set. This was done by selecting the best performing model from one set of experiments and using it as a default model for the next set of experiments. Only *CGN-larger* model does not follow this rule this model is altered on the architectural level rather than simply the level of hypeparameter tuning.

5.3.1.1 Experiments

All ten experiments are carried out with a batch size of 12 during training and online evaluation. The total duration of training for all models is approximately 30 epochs. A single scene sample is represented as 20000 points for the scene and 2000 points for the gripper, each sample has an auxiliary binary feature channel that defines whether the point belongs to the gripper or to the scene. For each batch during training and testing the ratio of samples with the ground truth class of 1 (colliding grasp) and 0 (valid grasp) is approximately $1:1$. The models are evaluated and ranked on the basis of their classification accuracy of the *ACRONYM-FORK01-COL* dataset last 1000 scenes (with the used batch size of 12 random views per scene this amounts to 12000 samples).

The first experiment used the default parameters described in 4.3. This model is denoted as *CGN-baseline*. Tables 5.4 - 5.6 provide an overview of the models that took part in the first evaluation stage: model name, parameters used during training that differ from the default ones.

5.3.1.2 Results

Table 5.7 summarizes the performance of the models by comparing their classification accuracies (ratio of correctly classified samples to the total number of samples), as well as precision and recall metrics for both classes.

With the absolute classification accuracies ranging from 0.709 to 0.873 across all three sets, a single best performing model from each set was chosen for the second stage of assessment, namely *CGN-pts↑-rad↓*, *CGN-pts↑-rad↓↓*, and *CGN-larger* from the first, the second and the third sets, respectively.

Name	Parameters
CGN-pts↑	1st SA number of points: 1024 2nd SA layer's number of points: 256
CGN-rad↓	1st SA radii list: $[0.05, 0.1, 0.2]$ 2nd SA layer's radii list: $[0.1, 0.2, 0.4]$
CGN-pts↑-rad↓	1st SA number of points: 1024 1st SA radii list: $[0.05, 0.1, 0.2]$ 2nd SA number of points: 256 2nd SA radii list: $[0.1, 0.2, 0.4]$

Table 5.4: Parameters used for the first set of experiments.

Name	Parameters
CGN-lr↑	1st SA number of points: 1024 1st SA radii list: $[0.05, 0.1, 0.2]$ 2nd SA layer's number of points: 256 2nd SA radii list: $[0.1, 0.2, 0.4]$ base learning rate: 0.002
CGN-decay↑	1st SA number of points: 1024 1st SA radii list: $[0.05, 0.1, 0.2]$ 2nd SA number of points: 256 2nd SA radii list: $[0.1, 0.2, 0.4]$ learning rate decay rate: 0.9
CGN-pts↑-rad↓↓	1st SA number of points: 1024 1st SA radii list: $[0.025, 0.05, 0.1]$ 2nd SA number of points: 256 2nd SA radii list: $[0.05, 0.1, 0.2]$

Table 5.5: Parameters used for the second set of experiments.

5.3.1.3 Discussion

One may notice that the best performing model in each set of experiments is outperforming the best model of the previous set. This means that the general

Name	Parameters
CGN-lr↑-decay↑	1st SA number of points: 1024 1st SA radii list: $[0.025, 0.05, 0.1]$ 2nd SA number of points: 256 2nd SA radii list: $[0.05, 0.1, 0.2]$ base learning rate: 0.002 learning rate decay rate: 0.9
CGN-lr-cosine	1st SA number of points: 1024 1st SA radii list: $[0.025, 0.05, 0.1]$ 2nd SA number of points: 256 2nd SA radii list: $[0.05, 0.1, 0.2]$ base learning rate: 0.002 cosine learning rate scheduling min learning rate: 5%
CGN-larger	1st SA number of points: 1024 1st SA radii list: $[0.01, 0.02, 0.4]$ 1st SA MLP lists: $[16, 16, 32], [32, 32, 64], [32, 48, 64]$ 1st SA point sample list: $[8, 16, 128]$ 2nd SA number of points: 512 2nd SA radii list: $[0.02, 0.04, 0.08]$ 2nd SA MLP lists: $[32, 32, 64], [64, 64, 128], [64, 96, 128]$ 2nd SA point sample list: $[16, 32, 128]$ 3rd SA number of points: 128 3rd SA radii list: $[0.04, 0.08, 0.16]$ 3rd SA MLP lists: $[64, 64, 128], [128, 128, 256], [128, 128, 256]$ 3rd SA point sample list: $[32, 64, 128]$ base learning rate: 0.002

Table 5.6: Parameters used for the third set of experiments.

lane of hyperparameter tuning was chosen well.

The biggest performance boost was achieved by reducing the local region search radii values of the set abstraction layers. The reason for such an improvement may

Set	Model	Accuracy	Precision (col./val.)	Recall (col./val.)
–	*CGN-baseline*	0.678	0.700/0.660	0.623/0.733
1	*CGN-pts*↑	0.709	0.711/0.706	0.703/0.714
1	*CGN-rad*↓	0.776	0.798/0.757	0.738/0.813
1	*CGN-pts*↑*-rad*↓	**0.811**	0.794/0.830	0.839/0.783
2	*CGN-lr*↑	0.798	0.796/0.799	0.800/0.800
2	*CGN-decay*↑	0.823	0.809/0.840	0.846/0.800
2	*CGN-pts*↑*-rad*↓↓	**0.856**	0.841/0.872	0.877/0.834
3	*CGN-lr*↑*-decay*↑	0.869	0.860/0.879	0.882/0.857
3	*CGN-lr-cosine*	0.859	0.868/0.850	0.847/0.871
3	*CGN-larger*	**0.873**	0.851/0.900	0.906/0.841

Table 5.7: Comparison of all the models trained during the first evaluation stage. Classification accuracy as well as precision and recall values (for both classes) are reported.

be the following: in the hardest cases of collision, the collision itself happens at an extremely small scale, thus it makes sense to reduce the radii to account for this. *CGN-larger*, for example, has the lowest radius set to 0.01 m which is the finger width of the gripper, and the biggest one to 0.16 which roughly approximates the length of the gripper. The best performing model in each set used reduced radii values when compared with the rest of the models in the same set.

Next comes the number of points sampled with FPS. The boost given by the increasing the number of points can be explained in a similar fashion: since collisions happen on an extremely small scale when compared to the rest of the scene, the overall performance may benefit from higher sampling density. When combined with finer radii value steps, the effect is amplified.

Tweaking the learning rate regimes, especially reducing the decay factor of the learning rate (in case of an exponential decay scheduler) slightly improved the performance. Cosine scheduler did not show any significant impact on performance.

CGN-larger is a special case, since it diverged from the strict hyperparameter tuning and instead had its architecture altered by introducing an additional set

abstraction layer. Local regions search radii became even finer, and the amount of points sampled in each local region changed, too. The fact that *CGN-larger* is the overall best performing model on the *ACRONYM-FORK01-COL* dataset may indicate that *Collision-GraspNet* architecture still has substantial potential for improvement via careful parameter tuning and even via making the model larger.

5.3.2 Evaluation on the GraspNet-1Billion dataset

The first evaluation stage produced three models that are best performing in their respective experiment set brackets. Next step is to evaluate the models on the predictions of the best performing *Contact-GraspNet* (see Table 5.2), which essentially are the predictions made on the real measurements. The reason to move forward with all three instead of a single best one is that since the second stage involves a domain gap (from classifying the synthetic scenes of *ACRONYM-FORK01-COL* to the real measurements of *GraspNet-1Billion*), thus the behavior of multiple *Contact-GraspNet* models should be observed during this stage to determine how different are the predictions used during this stage when compared to the ground truth proposals present in the *ACRONYM-FORK01-COL* dataset. The results produced by the models may give an insight of how different these scenes (and corresponding predicted grasp poses) are in terms of the classification difficulty.

5.3.2.1 Experiments

The first step is to compute the ground truth labels for the predicted grasp poses. In order to obtain them, all predictions are analytically labeled as colliding or valid in a mesh scene representation. The next step is to establish a baseline. *GraspNet* model-free collision manager (MFCM) evaluates all predictions on a corresponding point cloud representations of a partially observed scenes with default parameters as described in chapter 4.1.1.3. Then, all predictions are classified by *CGN-pts*↑*-rad*↓, *CGN-pts*↑*-rad*↓↓, and *CGN-larger* models. The ground truth class label distribution of the predictions made by the best performing *Contact-GraspNet* model on the test scenes of *GraspNet-1Billion* dataset is the following: for 1800424 predicted poses in total, 1136279 are valid (non-colliding), and 664145 are colliding.

5.3.2.2 Results

Table 5.8 reports *GraspNet-MFCM* classification results of the target predictions.

Actual \ Predicted	Colliding	Valid	Support	Precision	Recall
Colliding	497935	166210	664145	0.527	0.750
Valid	447584	688695	1136279	0.806	0.606

Table 5.8: *GraspNet-MFCM* classification results of the target grasp pose predictions.

Figure 5.5 provides a precision-recall curve for class 1 (colliding) of *CGN-pts↑-rad↓*, *CGN-pts↑-rad↓↓*, and *CGN-larger* models. Precision-recall values of *GraspNet-MFCM* are added for comparison.

Figure 5.6 provides a precision-recall curve for class 0 (non-colliding) of *CGN-pts↑-rad↓*, *CGN-pts↑-rad↓↓*, and *CGN-larger* models. Precision-recall values of *GraspNet-MFCM* are added for comparison.

5.3.2.3 Discussion

From the Figure 5.5 that only two *Collision-GraspNet* models are capable of outperforming the *GraspNet-MFCM* module, namely *CGN-pts↑-rad↓↓* and *CGN-larger*. *CGN-pts↑-rad↓* failed to outperform their analytical counterpart. The reason for this may be the following: the combination of the used query ball radii and point sampling density was not enough to capture the features that signify a collision case.

For the fixed recall value of 0.750, *CGN-larger* is capable of classification with precision of 0.583, and for the fixed precision value of 0.527 - of classification with recall of 0.816. This indicates that *Collision-GraspNet* is capable of outperforming the *GraspNet-MFCM*. For the case of the valid proposals as can be seen on Figure 5.6, *CGN-larger* shows better performance on a wide range of confidence thresholds when compared with the *GraspNet-MFCM*. Colliding class is more important in the task of collision classification, thus the optimal confidence threshold value is to be picked based on the precision-recall curve segment of the *CGN-larger* that outperforms the *GraspNet-MFCM* for the colliding class.

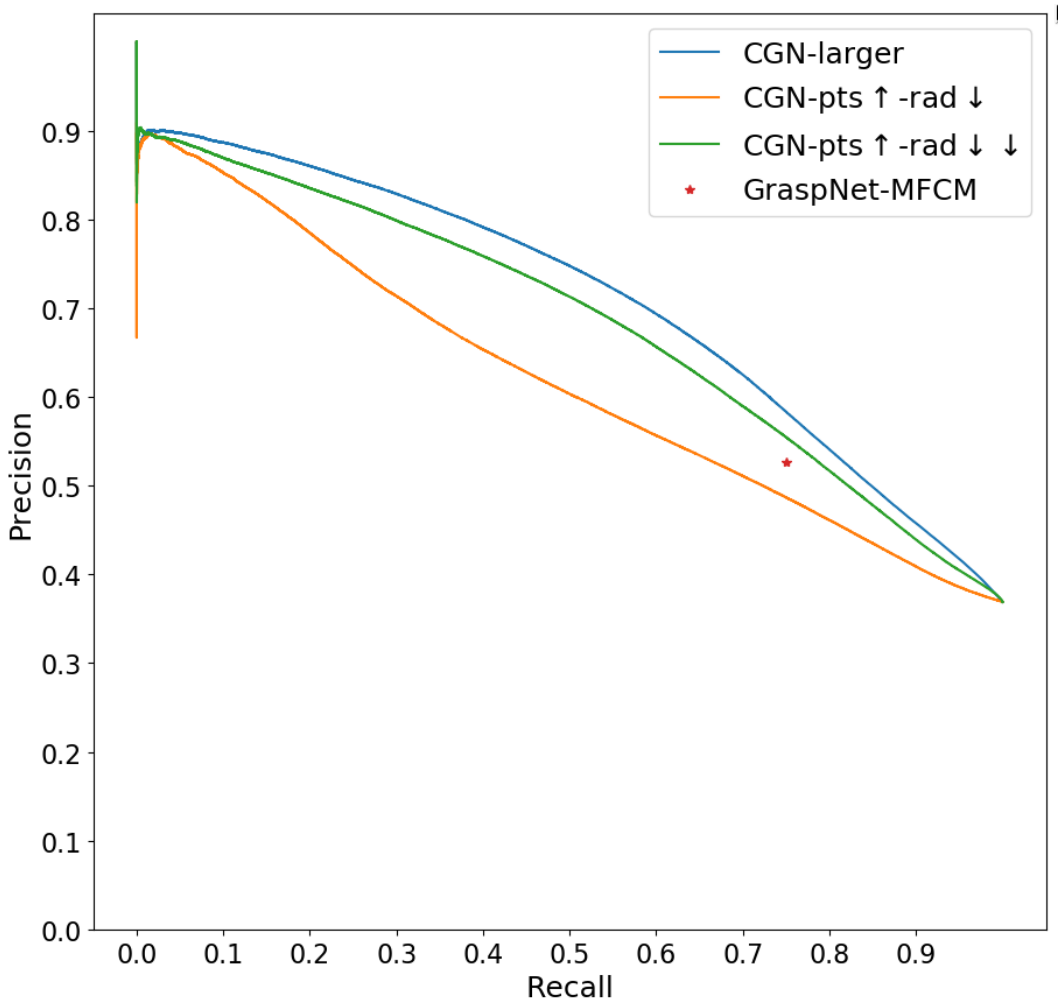


Figure 5.5: Precision-recall curve for class 1 (colliding). For the fixed recall value of 0.750 and the fixed precision value of 0.527, the *CGN-larger* model is capable of classification with precision of 0.583 and recall of 0.816, respectively.

Firstly, it is important to establish whether the higher recall or precision is more important when classifying the proposals. It is obvious, that having both precision and recall as high as possible (ideally both should be 1.0 for an ideal classificator), but usually a trade-off has to be made. One may argue that higher recall is more beneficial, since it intuitively means that the classifier is more efficient at detecting collisions, thus more colliding proposals can be refined. For example,

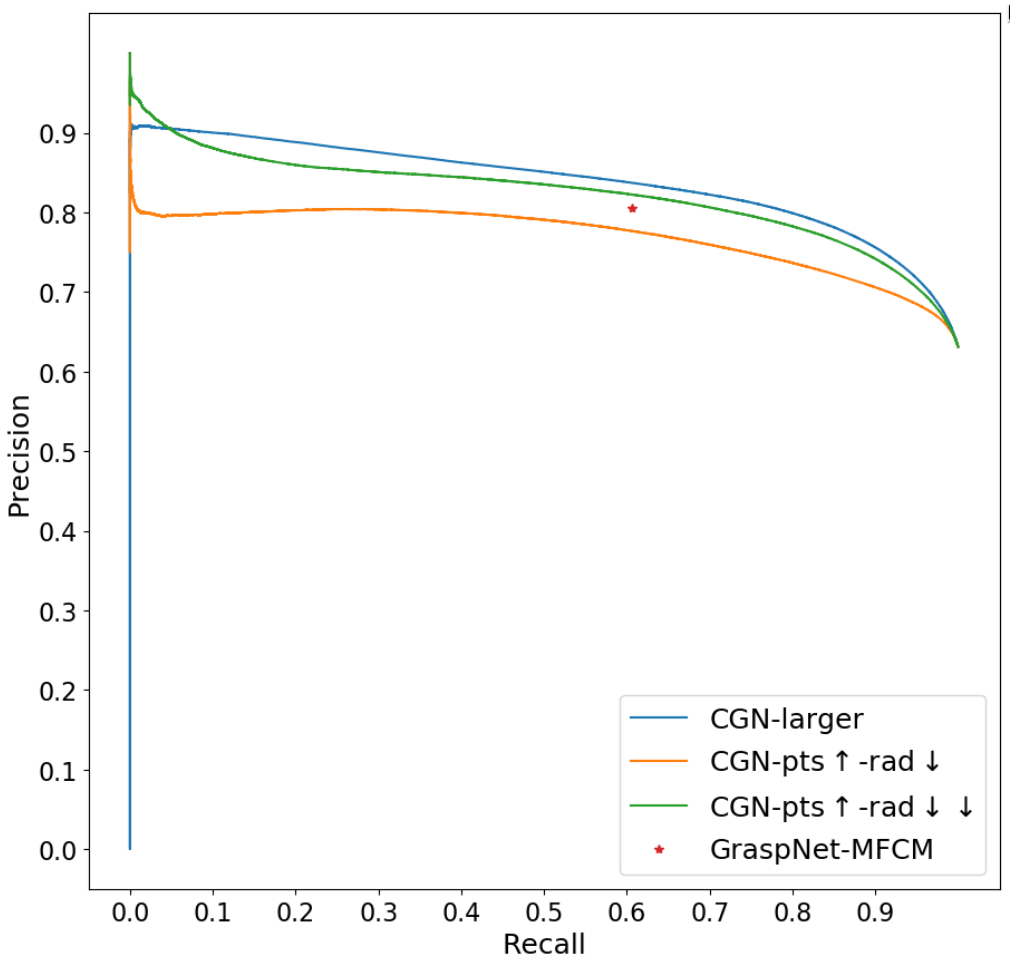


Figure 5.6: Precision-recall curve for class 0 (non-colliding).

the middle ground from the set of thresholds that lead to direct outperforming of the *GraspNet-MFCM* by both precision and recall metrics is the threshold of 0.046 which leads to the precision and recall values of 0.555 and 0.782 respectively against *GraspNet-MFCM* precision of 0.527 and recall of 0.750. This threshold leads to the weighted accuracy and weighted F1-score of 0.708 and 0.694, respectively, against 0.668 and 0.665 of *GraspNet-MFCM*. The corresponding threshold's value being so low can be explained by the fact that simulation-to-real domain gap between training data from the *ACRONYM-FORK01-COL* and test data from the *GraspNet-1Billion* datasets is present here, thus making classification more difficult. Additionally, the proposals that were classified during this evaluation stage do

not have that many self-evident collision cases (of direct collision with the visible part of the scene) that are present in the training data and mostly represent the hardest case of collision with the occluded part of the scene, further hindering the classification performance.

On the other hand, higher precision would mean that the consequential proposal refinement or rejection will be less "invasive" in the sense that it is less likely to affect the already valid proposals, thus such classifier is less likely to reduce the respective proposals scores according to some evaluation pipeline like the one described in 4.1.3, for example. In case of the precision, *Collision-GraspNet* can classify with considerably higher precision at the expense of recall, which would satisfy aforementioned requirement of minimal "invasiveness". Confidence threshold of 0.5, for example, produces the recall and precision values of 0.508 and 0.744, respectively. This, in its turn, produces weighted average accuracy of 0.703 the weighted F1-score of 0.741, against the weighted accuracy of 0.668 and the weighted F1-score of 0.665 of *GraspNet-MFCM*.

Ultimately, choosing the appropriate optimal confidence threshold value depends on the how much trust one is ready to put into the respective classification results, and what exactly is to happen with the proposals classified as colliding. For the next evaluation stage the confidence threshold of 0.5 was chosen, favoring classification precision over classification recall.

5.3.3 Grasp proposals refinement

After establishing that *Collision-GraspNet* is capable of standing toe-to-toe with its analytical counterpart in the task of collision classification, the final step of assessment aims at determining whether *Collision-GraspNet* is capable of reliably improving the grasp poses classified as colliding using the method described in chapter 4.3.

5.3.3.1 Experiments

In order achieve the goal of the final evaluation stage, the overall best performing model, namely *CGN-larger* yet again classifies the predictions used during the previous step of assessment, but this time each grasp pose prediction classified as colliding is attempted to be improved. Confidence threshold was set to 0.5.

Rotation and translation components of such grasps were sampled with the default values of 0.261799 radians and 0.02 m for rotation and translation, respectively. For each grasp pose, *Collision-GraspNet* attempted a perturbation for a maximum amount of 100 tries. If the sample (concatenated scene and gripper point clouds) of the new perturbed scene was classified by *Collision-GraspNet* as a valid case, the next colliding grasp was in line for perturbation. If network failed to successfully perturb the pose in the set amount of tries, this pose was omitted. The outputted perturbed grasp poses were evaluated based on two criteria: whether the pose was out of collision with the mesh representation of the full scene, and whether the perturbed grasp was still grasping an object in the scene.

For this step, only a subset of target predictions is used, namely predictions for scenes $100, 114, 129, 130, 144, 159, 160, 174$, and 189, that in total amass 145934 predictions, of which 50461 are in collision with a full scene mesh.

5.3.3.2 Results

Table 5.9 reports the predicted grasp poses perturbation results using *Collision-GraspNet* model: classification accuracy, amount of successfully perturbed grasps, the amount of successfully perturbed grasps that are no longer in collision with the scene mesh, the amount of successfully perturbed grasps that are still grasping an object in the scene, and the amount of successfully perturbed grasps that are fully valid (both non-colliding and grasping an object).

Figure 5.7 visualizes the original grasp poses and their perturbed versions, where red gripper color signifies that this pose (either original or perturbed) brings the gripper into a direct collision with the scene mesh, and green - that this pose is non-colliding. All perturbed grasp poses are grasping an object in the scene.

Classification of a single proposal takes on average around 0.1 sec, and *GraspNet-MFCM* takes around 0.08 sec per sample.

5.3.3.3 Discussion

Collision-GraspNet improved 51.5% of proposal poses it attempted to perturb in order to bring them out of collision. The number of successfully perturbed grasps that remained in a grasping pose in relation the object (96.7%) shows that the chosen rotation and translation boundary values were selected well, though the

Classification results	
Precision	0.729
Recall	0.531
Support	50461
Refinement results	
Attempted	36808
Successful	
– Total	34896(93.8%)
– Non-colliding	19131(54.8%)
– Grasping	33727(96.7%)
Valid	17962(51.5%)

Table 5.9: Classification results for class 1 (colliding) and corresponding perturbation results.

number of successfully perturbed grasps that remained in collision after 100 perturbation attempts may indicate that the composition of the scenes requires further boundary values tuning. Overall, taking into an account the reported classification precision value of 0.729 and the amount of successfully rescued proposals, it can be concluded that *Collision-GraspNet* is not only capable of reliable classification, but also of grasp proposal improvement in terms of collision.

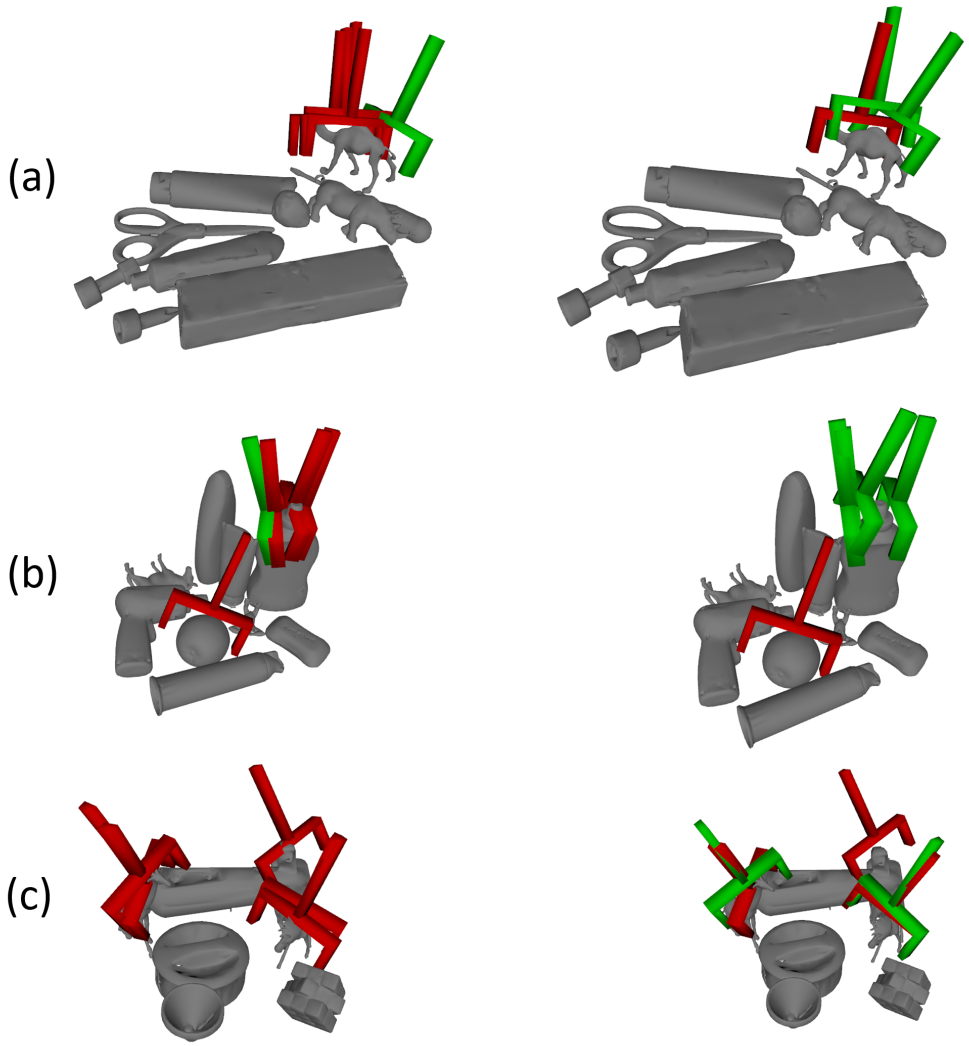


Figure 5.7: First 5 predicted grasp poses for scene 100 (a), 130 (b), and 160 (c) that are classified by the *Collision-GraspNet* as colliding: (left) original poses, (right) successfully perturbed poses in 100 tries. Color signifies the ground truth labels of the original and perturbed grasp poses (red color for colliding grasp, green - for non-colliding grasps).

6 Future steps

Following are a number of details and ideas that could be investigated to obtain a better in-depth understanding of the effectiveness of the proposed methods, as well as improve their performance.

Firstly, the persisting domain gap between the *ACRONYM* and the *GraspNet-1Billion* datasets and their respective derivative datasets could be addressed by using the modified *GraspNet-1Billion* real scene measurements for both training and testing of *Contact-GraspNet*. It would require processing all dataset grasp pose ground truth annotations to adopt the respective grasp representation, which would imply a significant computational effort.

While testing the modified mean curvature measure and generating the respective datasets, hard computational bottlenecks were encountered. The aforementioned processing of *GraspNet-1Billion* would not only eliminate the domain gap, but also remove the pool of poorly wired meshes that lead to botched training data embeddings. Additionally, having a single dataset would mean that estimating the optimal hyperparameters for a curvature-embedded dataset generation would be considerably easier.

Compared to *Contact-GraspNet*, *Collision-GraspNet* uses a similar data pipeline during both training and inference as well as a similar contact mapping step. Thus, the next logical step would be to train both *Contact-GraspNet* and *Collision-GraspNet* in an end-to-end fashion jointly. This would require having a dataset representing both valid and colliding ground truth grasp poses. Such dataset is *ACRONYM-FORK01-COL*, yet in the case of computing the aforementioned modified version of *GraspNet-1Billion*, colliding ground truth samples must be also included.

While the general composition of the chosen evaluation pipeline proved reasonable, certain parts of it are mostly hindering the overall suitability of this benchmark. The non-optional grasp NMS does not allow for direct evaluation of refined proposals

produced by the *Collision-GraspNet*, and the collision detection on the voxelized full scene is not particularly precise. Additionally, while a force-closure metric did prove itself a stable and reliable analytical grasp quality metric, it is still a rough approximation especially when taking into account the coarse friction coefficient grid employed in the chosen evaluation pipeline. Moreover, the pipeline could benefit from implementation optimization. Addressing these issues would require building a custom evaluation pipeline. One way would be to propose a pipeline that evaluates grasps in multiple stages on the levels of an individual grasp, an object, and a scene separately, thus giving a detailed insight into how certain properties of the predictions are affecting the overall scores on different levels. Also such a pipeline could employ an execution step in a simulated environment, for example using the FleX physics simulator, to complement the force-closure metric.

7 Conclusion

In this thesis, two approaches aimed at addressing the limitations imposed by the partial visibility and cluttered nature of a target scene during 6-DoF data-driven grasp pose estimation were proposed.

The first explores the possibility of embedding input data with differential geometrical shape information, namely the modified mean curvature measure, to improve the qualitative results of grasp estimation. During the corresponding experiments the benchmark baseline performance threshold was established by the *GraspNet* and *Contact-GraspNet* models and the resulting accuracy values were compared with *Contact-GraspNet* trained and tested on the curvature-embedded dataset derivatives of the *ACRONYM* and *GraspNet-1Billion* datasets. According to the results, *Contact-GraspNet* trained on these datasets failed to capture assumed geometrical patterns, presumably due to the poor quality of the embeddings in the train dataset.

The second method proposed a supervisor network architecture *Collision-GraspNet* that classifies grasp proposals with respect to collision with the scene, which also includes reasoning on the occluded parts of the scene by utilizing the ground truth grasp contact points, respective grasp poses and the gripper point cloud representation, and improves the invalid proposals via iterative pose sampling. During the corresponding experiments a baseline version of the *Collision-GraspNet* and nine variations of it were trained, where different lanes of hyperparameter tuning were explored. The models were evaluated on the test data split of the corresponding *ACRONYM* derivative dataset based on the classification accuracy, precision and recall values. Three models then were compared to the analytical grasp proposal filtering Model-free Collision Manager module employed by the *GraspNet* architecture by classifying the real proposals produced by the *Contact-GraspNet* during the testing of the first hypothesis. Two models outperformed the MFCM for such metrics as precision and recall for both colliding and non-colliding

classes. Additionally, one model is capable of classification with much higher precision, as well as with comparatively higher weighted F1-score and weighted average accuracy values for the target class of colliding proposals. This indicates that *Collision-GraspNet* is able to reason about the collisions of the proposed gripper pose with the occluded part of the scene. The best performing variation of the *Collision-GraspNet* was then evaluated on the subset of the same proposals, where the model's ability to refine the colliding poses was tested. The final experiment showed that *Collision-GraspNet* can not only reliable classify proposals with high precision, but also enhance around 50% of the proposals classified as colliding where the refined version is no longer in collision and still grasps an object in the scene.

List of Figures

List of Tables

Bibliography

[A.+18] Zeng A. et al. “Learning synergies between pushing and grasping with self-supervised deep reinforcement learning.” In: *International Conference on Intelligent Robots and Systems (IROS)* (2018), pp. 4238–4245.

[Ald+11] Aitor Aldoma et al. “CAD-model recognition and 6DOF pose estimation using 3D cues.” In: *2011 IEEE international conference on computer vision workshops (ICCV workshops)*. IEEE. 2011, pp. 585–592.

[Aok+19] Yasuhiro Aoki et al. “Pointnetlk: Robust & efficient point cloud registration using pointnet.” In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 7163–7172.

[BJR19] Gideon Billings and Matthew Johnson-Roberson. “Silhonet: An rgb method for 6d object pose estimation.” In: *IEEE Robotics and Automation Letters* 4.4 (2019), pp. 3727–3734.

[BK10] Jeannette Bohg and Danica Kragic. “Learning grasping points with shape context.” In: *Robotics and Autonomous Systems* 58.4 (2010), pp. 362–377.

[BTVG06] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. “Surf: Speeded up robust features.” In: *European conference on computer vision*. Springer. 2006, pp. 404–417.

[BWL20] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. “Yolov4: Optimal speed and accuracy of object detection.” In: *arXiv preprint arXiv:2004.10934* (2020).

[Cai+19] Junhao Cai et al. "Metagrasp: Data efficient grasping by affordance interpreter network." In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 4960–4966.

[Cal+17] Berk Calli et al. "Yale-CMU-Berkeley dataset for robotic manipulation research." In: *The International Journal of Robotics Research* 36.3 (2017), pp. 261–268.

[Che+18] Liang-Chieh Chen et al. "Masklab: Instance segmentation by refining object detection with semantic and direction features." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 4013–4022.

[Che+19] Xinlei Chen et al. "Tensormask: A foundation for dense object segmentation." In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 2061–2069.

[Che+20] Hao Chen et al. "BlendMask: Top-down meets bottom-up for instance segmentation." In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 8573–8581.

[CXV18] Fu-Jen Chu, Ruinian Xu, and Patricio A Vela. "Real-world multiobject, multigrasp detection." In: *IEEE Robotics and Automation Letters* 3.4 (2018), pp. 3355–3362.

[D.+18] Kalashnikov D. et al. "Scalable deep reinforcement learning for vision-based robotic manipulation." In: *Conference on Robot Learning* (2018), pp. 651–673.

[Dai+16] Jifeng Dai et al. "R-fcn: Object detection via region-based fully convolutional networks." In: *arXiv preprint arXiv:1605.06409* (2016).

[DDC18] Amaury Depierre, Emmanuel Dellandréa, and Liming Chen. *Jacquard: A Large Scale Dataset for Robotic Grasp Detection*. 2018. arXiv: `1803.11469 [cs.RO]`.

[DDC20] Amaury Depierre, Emmanuel Dellandréa, and Liming Chen. "Optimizing correlated graspability score and grasp regression for better grasp prediction." In: *arXiv preprint arXiv:2002.00872* (2020).

[Dom+14] Yukiyasu Domae et al. "Fast graspability evaluation on single depth maps for bin picking with general grippers." In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2014, pp. 1997–2004.

[DP73] David H Douglas and Thomas K Peucker. "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature." In: *Cartographica: the international journal for geographic information and geovisualization* 10.2 (1973), pp. 112–122.

[Du+20] Guoguang Du et al. "Vision-based robotic grasping from object localization, object pose estimation to grasp estimation for parallel grippers: a review." In: *Artificial Intelligence Review* (2020).

[EMF19] Clemens Eppner, Arsalan Mousavian, and Dieter Fox. *A Billion Ways to Grasp: An Evaluation of Grasp Sampling Schemes on a Dense, Physics-based Grasp Data Set*. 2019. arXiv: `1912.05604 [cs.RO]`.

[EMF20] Clemens Eppner, Arsalan Mousavian, and Dieter Fox. *ACRONYM: A Large-Scale Grasp Dataset Based on Simulation*. 2020. arXiv: `2011.09584 [cs.RO]`.

[Eng+20] Francis Engelmann et al. "3d-mpa: Multi-proposal aggregation for 3d semantic instance segmentation." In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 9031–9040.

[Fan+20] Zhibo Fan et al. "Fgn: Fully guided network for few-shot instance segmentation." In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 9172–9181.

[Fan+20] H. S. Fang et al. "GraspNet-1Billion: A Large-Scale Benchmark for General Object Grasping." In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 11441–11450.

[FB81] Martin A Fischler and Robert C Bolles. "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography." In: *Communications of the ACM* 24.6 (1981), pp. 381–395.

[FF+96] Andrew W Fitzgibbon, Robert B Fisher, et al. *A buyer's guide to conic fitting*. Citeseer, 1996.

[FMT18] Peter R Florence, Lucas Manuelli, and Russ Tedrake. "Dense object nets: Learning dense visual object descriptors by and for robotic manipulation." In: *arXiv preprint arXiv:1806.08756* (2018).

[Fro+04] Andrea Frome et al. "Recognizing objects in range data using regional point descriptors." In: *European conference on computer vision*. Springer. 2004, pp. 224–237.

[GGF20] Yulong Wang Xi-aolin Hu Jianwei Zhang Ge Gao Mikko Lauri and Simone Frintrop. "6d object pose regression via supervised learning on point clouds." In: *arXiv* (2020).

[GH97] Michael Garland and Paul S Heckbert. "Surface simplification using quadric error metrics." In: *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*. 1997, pp. 209–216.

[Gir+14] Ross Girshick et al. "Rich feature hierarchies for accurate object detection and semantic segmentation." In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014, pp. 580–587.

[Gir15] Ross Girshick. "Fast r-cnn." In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1440–1448.

[Han+20] Lei Han et al. "Occuseg: Occupancy-aware 3d instance segmentation." In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 2940–2949.

[HBM20] Tomas Hodan, Daniel Barath, and Jiri Matas. "EPOS: estimating 6D pose of objects with symmetries." In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 11703–11712.

[HDN19] Ji Hou, Angela Dai, and Matthias Nießner. "3d-sis: 3d semantic instance segmentation of rgb-d scans." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 4421–4430.

[He+17] Kaiming He et al. "Mask r-cnn." In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2961–2969.

[HSG18] Jingwei Huang, Hao Su, and Leonidas Guibas. *Robust Watertight Manifold Surface Generation Method for ShapeNet Models*. 2018. arXiv: 1802.01698 [cs.CG].

[Hu+19] Yinlin Hu et al. "Segmentation-driven 6d object pose estimation." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 3385–3394.

[IP17] Timothy Lillicrap Roland Hafner Gabriel Barth-Maron Matej Vecerik Thomas Lampe Yuval Tassa Tom Erez Martin Riedmille Ivaylo Popov Nicolas Heess. "Data-efficient deep reinforcement learning for dexterous manipulation." In: *arXiv)* (2017).

[Jia+13] Huaizu Jiang et al. "Salient object detection: A discriminative regional feature integration approach." In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2013, pp. 2083–2090.

[JMS11a] Yun Jiang, Stephen Moseson, and Ashutosh Saxena. "Efficient grasping from rgbd images: Learning using a new rectangle representation." In: *2011 IEEE International conference on robotics and automation*. IEEE. 2011, pp. 3304–3311.

[JMS11b] Yun Jiang, Stephen Moseson, and Ashutosh Saxena. "Efficient grasping from RGBD images: Learning using a new rectangle representation." In: *2011 IEEE International Conference on Robotics and Automation*. 2011, pp. 3304–3311.

[Joh97] Andrew E Johnson. "Spin-images: a representation for 3-D surface matching." In: (1997).

[KBS15] D. Kappler, J. Bohg, and S. Schaal. "Leveraging big data for grasp planning." In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. 2015, pp. 4304–4311.

[Lev+16] Sergey Levine et al. *Learning Hand-Eye Coordination for Robotic Grasping with Deep Learning and Large-Scale Data Collection*. 2016. arXiv: 1603.02199 [cs.LG].

[LF19] Chen Liu and Yasutaka Furukawa. “Masc: Multi-scale affinity with sparse convolution for 3d instance segmentation.” In: *arXiv preprint arXiv:1902.04478* (2019).

[LH16] Nian Liu and Junwei Han. “Dhsnet: Deep hierarchical saliency network for salient object detection.” In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 678–686.

[LHY18] Nian Liu, Junwei Han, and Ming-Hsuan Yang. “Picanet: Learning pixel-wise contextual attention for saliency detection.” In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 3089–3098.

[Lia+19a] Hongzhuo Liang et al. “Pointnetgpd: Detecting grasp configurations from point sets.” In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 3629–3635.

[Lia+19b] Ming Liang et al. “Multi-task multi-sensor fusion for 3d object detection.” In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 7345–7353.

[Lin+17] Tsung-Yi Lin et al. “Feature pyramid networks for object detection.” In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 2117–2125.

[Lin20] Haibin Ling. “Cross-modal weighting network for RGB-D salient object detection.” In: (2020).

[Liu+19] Fuchang Liu et al. “Recovering 6D object pose from RGB indoor image based on two-stage detection network with multi-task loss.” In: *Neurocomputing* 337 (2019), pp. 15–23.

[LLS15] Ian Lenz, Honglak Lee, and Ashutosh Saxena. “Deep learning for detecting robotic grasps.” In: *The International Journal of Robotics Research* 34.4-5 (2015), pp. 705–724.

[Low99] David G Lowe. “Object recognition from local scale-invariant features.” In: *Proceedings of the seventh IEEE international conference on computer vision*. Vol. 2. Ieee. 1999, pp. 1150–1157.

[LVK19] Jens Lundell, Francesco Verdoja, and Ville Kyrki. "Robust grasp planning over uncertain shape completions." In: *arXiv preprint arXiv:1903.00645* (2019).

[LYC20] Xibai Lou, Yang Yang, and Changhyun Choi. "Learning to generate 6-dof grasp poses with reachability awareness." In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 1532–1538.

[MA04] Andrew T Miller and Peter K Allen. "Graspit! a versatile simulator for robotic grasping." In: *IEEE Robotics & Automation Magazine* 11.4 (2004), pp. 110–122.

[Mac+14] Miles Macklin et al. "Unified Particle Physics for Real-Time Applications." In: *ACM Trans. Graph.* 33.4 (July 2014).

[Mah+17] Jeffrey Mahler et al. "Dex-Net 2.0: Deep Learning to Plan Robust Grasps with Synthetic Point Clouds and Analytic Grasp Metrics." In: July 2017.

[MCL18] Douglas Morrison, Peter Corke, and Jürgen Leitner. "Closing the loop for robotic grasping: A real-time, generative grasp synthesis approach." In: *arXiv preprint arXiv:1804.05172* (2018).

[MEF19a] Arsalan Mousavian, Clemens Eppner, and Dieter 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.

[MEF19b] Arsalan Mousavian, Clemens Eppner, and Dieter Fox. *6-DOF GraspNet: Variational Grasp Generation for Object Manipulation*. May 2019.

[Mer+20] Mark Van der Merwe et al. "Learning continuous 3d reconstructions for geometrically aware grasping." In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 11516–11522.

[MG17] Jeffrey Mahler and Ken Goldberg. "Learning deep policies for robot bin picking by simulating robust grasping sequences." In: *Conference on robot learning*. PMLR. 2017, pp. 515–524.

[Mil+03] Andrew T Miller et al. "Automatic grasp planning using shape primitives." In: *2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422)*. Vol. 2. IEEE. 2003, pp. 1824–1829.

[MLS94] Murray Richard M., Zexiang Li, and Sastry S. Shankar. *A mathematical introduction to robotic manipulation*. CRC Press, 1994.

[MS21] Rudolph Triebel Dieter Fox Martin Sundermeyer Arsalan Mousavian. "Contact-GraspNet: Efficient 6-DoF Grasp Generationin Cluttered Scenes." In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021.

[MSG20] Stefan Milz Christian Tobias Witt Martin Simon Kai Fischer and Horst-Michael Gross. "Stickypillars: Robust feature matching on point clouds usinggraph neural networks." In: *arXiv* (2020).

[Mur+20] Adithyavairavan Murali et al. "6-dof grasping for target-driven object manipulation in clutter." In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 6232–6238.

[Naj+20] Mahyar Najibi et al. "Dops: learning to detect 3d objects and predict their 3d shapes." In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 11913–11922.

[Ngu88] Van-Duc Nguyen. "Constructing Force- Closure Grasps." In: *The International Journal of Robotics Research* 7.3 (1988), pp. 3–16.

[Pan+20] Youwei Pang et al. "Hierarchical dynamic filtering network for RGB-D salient object detection." In: *arXiv preprint arXiv:2007.06227* (2020).

[Pas+17a] Andreas ten Pas et al. *Grasp Pose Detection in Point Clouds*. 2017. arXiv: `1706.09911 [cs.RO]`.

[Pas+17b] Andreas ten Pas et al. "Grasp pose detection in point clouds." In: *The International Journal of Robotics Research* 36.13-14 (2017), pp. 1455–1473.

[Pen+14] Houwen Peng et al. "Rgbd salient object detection: a benchmark and algorithms." In: *European conference on computer vision*. Springer. 2014, pp. 92–109.

[PG15] Lerrel Pinto and Abhinav Gupta. *Supersizing Self-supervision: Learning to Grasp from 50K Tries and 700 Robot Hours*. 2015. arXiv: 1509.06825 [cs.LG].

[PG16] Lerrel Pinto and Abhinav Gupta. "Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours." In: *2016 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2016, pp. 3406–3413.

[Pia+19] Yongri Piao et al. "Depth-induced multi-scale recurrent attention network for saliency detection." In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 7254–7263.

[PP15] Andreas ten Pas and Robert Platt. "Using geometry to detect grasps in 3d point clouds." In: *arXiv preprint arXiv:1501.03100* (2015).

[PPV20] Timothy Patten, Kiru Park, and Markus Vincze. "Dgcm-net: dense geometrical correspondence matching network for incremental experience-based robotic grasping." In: *arXiv preprint arXiv:2001.05279* (2020).

[Qi+17a] Charles R Qi et al. "Pointnet++: Deep hierarchical feature learning on point sets in a metric space." In: *arXiv preprint arXiv:1706.02413* (2017).

[Qi+17b] Charles R Qi et al. "Pointnet: Deep learning on point sets for 3d classification and segmentation." In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 652–660.

[Qi+18] Charles R Qi et al. "Frustum pointnets for 3d object detection from rgb-d data." In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 918–927.

[Qi+19] Qi Qi et al. "Multi-scale capsule attention-based salient object detection with multi-crossed layer connections." In: *2019 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE. 2019, pp. 1762–1767.

[Qin+20] Yuzhe Qin et al. "S4g: Amodal single-view single-shot se (3) grasp detection in cluttered scenes." In: *Conference on robot learning*. PMLR. 2020, pp. 53–65.

[RA15] Joseph Redmon and Anelia Angelova. "Real-time grasp detection using convolutional neural networks." In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2015, pp. 1316–1322.

[RBB09] Radu Bogdan Rusu, Nico Blodow, and Michael Beetz. "Fast point feature histograms (FPFH) for 3D registration." In: *2009 IEEE international conference on robotics and automation*. IEEE. 2009, pp. 3212–3217.

[RD05] Edward Rosten and Tom Drummond. "Fusing points and lines for high performance tracking." In: *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*. Vol. 2. Ieee. 2005, pp. 1508–1515.

[Red+16] Joseph Redmon et al. "You only look once: Unified, real-time object detection." In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 779–788.

[Ren+15] Shaoqing Ren et al. "Faster r-cnn: Towards real-time object detection with region proposal networks." In: *arXiv preprint arXiv:1506.01497* (2015).

[RF17] Joseph Redmon and Ali Farhadi. "YOLO9000: better, faster, stronger." In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 7263–7271.

[RF18] Joseph Redmon and Ali Farhadi. "Yolov3: An incremental improvement." In: *arXiv preprint arXiv:1804.02767* (2018).

[RIP17] M. Gomrokchi R. Islam P. Henderson and D. Precup. "Reproducibility of benchmarked deep reinforcement learning tasks for continuous control." In: *arXiv)* (2017).

[Rub+11] Ethan Rublee et al. "ORB: An efficient alternative to SIFT or SURF." In: *2011 International conference on computer vision*. Ieee. 2011, pp. 2564–2571.

[RVDH05] Tahir Rabbani and Frank Van Den Heuvel. "Efficient hough transform for automatic detection of cylinders in point clouds." In: *Isprs Wg Iii/3, Iii/4* 3 (2005), pp. 60–65.

[Saj+20] Shreeyak Sajjan et al. “Clear Grasp: 3D Shape Estimation of Transparent Objects for Manipulation.” In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 3634–3642.

[Sar+19] Vinit Sarode et al. “PCRNet: Point cloud registration network using PointNet encoding.” In: *arXiv preprint arXiv:1908.07906* (2019).

[SCH15] Manolis Savva, Angel X Chang, and Pat Hanrahan. “Semantically-enriched 3D models for common-sense knowledge.” In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2015, pp. 24–31.

[Shi+20] Shaoshuai Shi et al. “Pv-rcnn: Point-voxel feature set abstraction for 3d object detection.” In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 10529–10538.

[Shi96] K. Shimoga. “Robot Grasp Synthesis Algorithms: A Survey.” In: *International Journal of Robotic Research* (1996), pp. 230–266.

[SR20] Weijing Shi and Raj Rajkumar. “Point-gnn: Graph neural network for 3d object detection in a point cloud.” In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 1711–1719.

[SSH20] Chen Song, Jiaru Song, and Qixing Huang. “Hybridpose: 6d object pose estimation under hybrid representations.” In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 431–440.

[Sun+18] Martin Sundermeyer et al. “Implicit 3d orientation learning for 6d object detection from rgb images.” In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 699–715.

[SWL19] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. “Pointrcnn: 3d object proposal generation and detection from point cloud.” In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 770–779.

[tC06] Gatzke t. and Grimm C. “Estimating curvature on triangular meshes.” In: *International Journal of Shape Modelingy* 12.1 (2006).

[Tia+19a] Hao Tian et al. “Transferring grasp configurations using active learning and local replanning.” In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 1622–1628.

[Tia+19b] Zhi Tian et al. “Fcos: Fully convolutional one-stage object detection.” In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 9627–9636.

[Tob+18] Joshua Tobin et al. *Domain Randomization and Generative Models for Robotic Grasping*. 2018. arXiv: `1710.06425 [cs.RO]`.

[Tos+20] Tarik Tosun et al. “Robotic grasping through combined image-based grasp proposal and 3d reconstruction.” In: *arXiv preprint arXiv:2003.01649* (2020).

[Var+17] Jacob Varley et al. “Shape completion enabled robotic grasping.” In: *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE. 2017, pp. 2442–2447.

[VMT17] Matthew Veres, Medhat Moussa, and Graham W. Taylor. *An Integrated Simulator and Dataset that Combines Grasping and Vision for Deep Learning*. 2017. arXiv: `1702.02103 [cs.RO]`.

[VPB19] Mohit Vohra, Ravi Prakash, and Laxmidhar Behera. “Real-time grasp pose estimation for novel objects in densely cluttered environment.” In: *2019 28th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*. IEEE. 2019, pp. 1–6.

[Wan+12] Zhe Wang et al. “Real-time plane segmentation and obstacle detection of 3D point clouds for indoor scenes.” In: *European Conference on Computer Vision*. Springer. 2012, pp. 22–31.

[Wan+18] Shaoxiong Wang et al. “3d shape perception from monocular vision, touch, and shape priors.” In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 1606–1613.

[Wan+19] Chen Wang et al. “Densefusion: 6d object pose estimation by iterative dense fusion.” In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 3343–3352.

[Wan+20] Chen Wang et al. “6-pack: Category-level 6d pose tracker with anchor-based keypoints.” In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 10059–10066.

[Won+17] Jay M Wong et al. “Segicp: Integrated deep semantic segmentation and pose estimation.” In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2017, pp. 5784–5789.

[WS19a] Yue Wang and Justin M Solomon. “Deep closest point: Learning representations for point cloud registration.” In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 3523–3532.

[WS19b] Yue Wang and Justin M Solomon. “Prnet: Self-supervised learning for partial-to-partial registration.” In: *arXiv preprint arXiv:1910.12240* (2019).

[XAJ18] Danfei Xu, Dragomir Anguelov, and Ashesh Jain. “Pointfusion: Deep sensor fusion for 3d bounding box estimation.” In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 244–253.

[Xie+20a] Enze Xie et al. “Polarmask: Single shot instance segmentation with polar representation.” In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 12193–12202.

[Xie+20b] Qian Xie et al. “Mlcvnet: Multi-level context votenet for 3d object detection.” In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 10447–10456.

[Yan+18] Xinchen Yan et al. “Learning 6-dof grasping interaction via deep geometry-aware 3d representations.” In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 3766–3773.

[Yi+19] Li Yi et al. “Gspn: Generative shape proposal network for 3d instance segmentation in point cloud.” In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 3947–3956.

[YML18] Yan Yan, Yuxing Mao, and Bo Li. "Second: Sparsely embedded convolutional detection." In: *Sensors* 18.10 (2018), p. 3337.

[Yu+20] Xin Yu et al. "6dof object pose estimation via differentiable proxy voting loss." In: *arXiv preprint arXiv:2002.03923* (2020).

[Zen+17a] Andy Zeng et al. "3dmatch: Learning local geometric descriptors from rgb-d reconstructions." In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 1802–1811.

[Zen+17b] Andy Zeng et al. "Multi-view self-supervised deep learning for 6d pose estimation in the amazon picking challenge." In: *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2017, pp. 1386–1383.

[Zen+18] Andy Zeng et al. "Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching." In: *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2018, pp. 3750–3757.

[Zha+15] Rui Zhao et al. "Saliency detection by multi-context deep learning." In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 1265–1274.

[Zha+16] Jianming Zhang et al. "Unconstrained salient object detection via proposal subset optimization." In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 5733–5742.

[Zha+19] Hanbo Zhang et al. *ROI-based Robotic Grasp Detection for Object Overlapping Scenes*. 2019. arXiv: `1808.10313 [cs.RO]`.

[Zha+20a] Feihu Zhang et al. "Instance segmentation of lidar point clouds." In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 9448–9455.

[Zha+20b] Binglei Zhao et al. "Regnet: region-based grasp network for single-shot grasp detection in point clouds." In: *arXiv preprint arXiv:2002.12647* (2020).

[ZT18] Yin Zhou and Oncel Tuzel. “Voxelnet: End-to-end learning for point cloud based 3d object detection.” In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 4490–4499.