Tampereen yliopisto

Toni Rossi

# DEEP LEARNING FOR ROBOT GRASPING

Tekniikan ja luonnontieteiden tiedekunta
Kandidaatintyö
Tarkastaja: Roel Pieters
Maaliskuu 2024

# TIIVISTELMÄ

Toni Rossi: Syväoppiminen robotin tarttumiselle
Kandidaatintyö
Tampereen yliopisto
Teknisten tieteiden kandidaatin tutkinto-ohjelma
Maaliskuu 2024

---

Robottikäden tarttuminen erilaisiin esineisiin on vaikea ongelma ratkaistavaksi. Tätä vaikeuttaa tartuttavan esineen muoto ja sen asento, tarttumiseen valittu päätetyökalu ja tiedonkeruu. Syväoppimista on hyödynnetty paljon robotin tarttumisessa, mikä tekee robotille esineiden tarttumisen oppimisen alusta loppuun mahdolliseksi. Syväoppimista hyödynnetään siten, että sillä luodaan neuroverkkoja tarttumisista kerätyn datan analysointiin. Tämän työn tarkoituksena oli tutkia aihetta kirjallisuuskatsauksena tarttumisen huippuluokkaan ja vertailla eri tarttumismalleja keskenään.

Robotin tarttumisen havaitseminen tyypillisesti saadaan aikaiseksi RGB-D-antureilla (Red Green Blue-Depth), jotka ottavat tavallisia RGB-kuvia sekä myös syvyyskuvia, jotka sitten muunnetaan pistepilvikuviksi 3D-tiloissa. Asennon arviointi tehdään hyödyntäen kuutta vapausastetta (engl. 6-DOF, Six-Degrees-of-Freedom), millä simuloidaan ihmisen kättä täydellä 3D-liikkeellä. Tutkimusten mukaan nämä kaksi seikkaa ovat perusta tarttumismalleille, jotka hyödyntävät syväoppimista. Kuitenkin 6-DOF:in tilalla saattoi olla yksi ylimääräinen vapausaste joidenkin mallien kohdalla.

Tarttumismallit ovat tietokoneohjelmistoja roboteille, mitkä hyödyntävät syväoppimista tarttumisissa. Tässä työssä tarkasteltiin kuutta eri tarttumismallia hyödyntämällä niiden kehittäjien dokumentteja kyseisistä malleista. 6-DOF GraspNet ja FFB6D (Full Flow Bidirectional fusion network for 6D) turvautuvat pistepilvikuvien analysointiin, mutta niillä on paljon muita eroavaisuuksia. ROI-GD (Region of Interest Grasp Detection) ja GR-ConvNet (Generative Residual Convolutional Neural Network) analysoivat kuvia eri moduulien kautta, jotka toimivat yhtäaikaisesti. Näiden kahden tarttumismallin välillä oli eniten samankaltaisuuksia muihin verrattuna. Dexterity-Network 4.0 kouluttaa tietynlaisia tarttumisvälineitä tarttuakseen tehokkaasti analyyttisillä malleilla ja GraspNet 1Billion on valtava dataverkko erilaisille esineille, missä on yli miljardi tarttumisasentoa erilaisille skenaarioille. Nämä kaksi tarttumismallia erosivat eniten muista tarkastelluista malleista. Tutkimuksen mukaan fyysiset kokeet eivät ole aina tarpeellisia tarttumismallien kehityksessä, koska pelkillä simulaatioillakin voi kouluttaa tarttumismalleja hyvin.

Mitä tulee robotin tarttumisen ja sen syväoppimisen kehittämiseen, tietokoneohjelmisto robottien tarttumisoperaatioihin vaatii lisää kehitystä, koska robottien kouluttaminen vie aikaa ja resursseja, ohjelmiston käytölle ei välttämättä ole taattua tukea ja ohjelmisto ei edes välttämättä toimi monille roboteille. Tämän lisäksi robotin tarttuminen vaikuttanee nyt turvautuvan paljon RGB-D-antureiden käyttöön, mutta tulevaisuudessa saattaa olla jotain tarkempaakin, kuin näiden antureiden ottamat syvyyskuvat.

Avainsanat: syväoppiminen, robotin tarttuminen, tarttumismalli, 6-DOF, RGB-D

# ABSTRACT

Toni Rossi: Deep Learning for Robot Grasping
Bachelor's Degree
Tampere University
Degree Programme for Bachelor of Technical Sciences
March 2024

---

Robot grasping is a difficult problem to solve. What makes it difficult includes the graspable object's type, its pose, the end-effector used in grasping, and data collection. In robot grasping, deep learning makes it possible for the robot to learn how to grasp objects in an end-to-end manner. Deep learning is utilized by making neural networks from generated grasps to analyze collected data. This paper was meant to study this topic as a literature review about its state of the art and compare different grasping models with each other.

Robot grasp detections are done typically by using RGB-D (Red Green Blue-Depth) sensors that take a regular RGB image, and a depth image that is converted to a point cloud image in a 3D space. For full 3D movement to simulate a human hand, pose estimation is done with 6-DOF (Six-Degrees-of-Freedom) grasping. Grasping models are software for robots that perform grasps by using deep learning. According to this paper's findings, these two factors are the basis for grasping models that make use of deep learning. However, sometimes in place of 6-DOF there could have been one additional degree of freedom in some models.

Grasping models are computer programs for robots which utilize deep learning in robot grasping. In this paper, six grasping models were covered by utilizing their developers' documents for said models. 6-DOF GraspNet and FFB6D (Full Flow Bidirectional fusion network for 6D) rely heavily on point cloud analysis, but they have many other differences. ROI-GD (Region of Interest Grasp Detection) and GR-ConvNet (Generative Residual Convolutional Neural Network) analyze images by running them through different modules that work in tandem. These two grasping models had the most similarities when they were compared to others. Dexterity-Network 4.0 trains specific sets of grippers to grasp efficiently with analytic models, and GraspNet 1Billion is a massive datanet for different objects with over one billion grasp poses for different scenarios. These two grasping models differed the most from the other models that were covered. According to this paper's findings, physical tests are not always necessary when developing grasping models because they could also be trained well with only simulations.

When it comes to further developing robot grasping and its deep learning, the software for robots to perform grasps needs more work because it takes time and resources to teach robots, there might not be any guaranteed support for its use, and it might not even work for many robots at all. Also, robot grasping seems to rely now heavily on RGB-D sensors, however in the future, there might be something more accurate than the depth images these sensors take.

Keywords: deep learning, robot grasping, grasping model, 6-DOF, RGB-D

# CONTENTS

# INTRODUCTION

Robot grasping, the act of a robot arm grasping an object with a gripper on its other end to move it to a desirable goal is a challenging problem, one that has seen many different methods that try to find a way to implement this operation. What makes it challenging are its various key components, such as a graspable object's type and pose, the gripper type used in grasping, and data collection for successful grasps. An example of a robot arm is presented in figure 1. Data collection, and its utilization is particularly difficult, which is the foundation for various methods trying to achieve robot grasping. Robot grasping has however evolved to a point where deep learning is utilized to achieve reliably successful grasps. Nowadays different researchers make use of deep learning to improve their methods, as the success rate of grasping has shown by these methods. They develop algorithms that interpret data in their own unique ways, which they can even make use of later when creating new methods for robot grasping.


***Figure 1.*** *An example of a robot industrial arm, developed by ABB.* [19, p. 1]

In this paper, six different methods for robot grasping using deep learning have been chosen for analysis, and will be covered briefly; how they work, how they are trained, what experiments were done with them etc. After that they will be compared with each other to see what differences and similarities they have in their various aspects. Before any of that however, robot grasping and its deep learning will be explained briefly to help the reader understand the chosen six different methods; what steps there are in a grasping process, and explaining some terminology related to robot grasping. These methods, or models as their developers call them are: 6-DOF GraspNet, Dex-Net 4.0, GraspNet-1Billion, FFB6D, ROI-GD, and GR-ConvNet. They are all relatively recent models, which should give the reader a look at the state of the art.

# 1. OVERVIEW OF ROBOT GRASPING

To understand how to utilize the different methods for robot grasping, the various grasping steps and important parts need to be explained, which when working together completes the task for a robot to grasp an object. Deep learning in robot grasping makes it possible for the robot to learn a grasping model in an end-to-end manner, which means that the robot does not need anything from a third party, and instead learns it itself. This way it can improve its ability to grasp objects. [1, p. 2] Deep learning is also defined as the science, in which large artificial neural networks are created and developed, also known as deep neural networks (DNN) [2, p. 821]. The different methods using deep learning all implement the grasping steps and important parts in their own ways in their own models, although with some of the same aspects as the others have.

Robot grasping consists of the following steps: grasp detection, object pose estimation, motion planning, grasping positioning, and the grasping itself. These steps can be achieved with different methods. Robot grasping also includes some important parts, such as the gripper, the object itself and its features, and data collecting to be utilized through deep learning. The data collection to be used in deep neural network development can be time consuming if there are no object models available, or if there is no suitable imaging system available. Models that use deep neural networks are also very demanding regarding resources, which means that many small enterprises can't possibly even use them. The models' sizes can't be too big, they need to be fast, they must have a low latency, and they need to be accurate enough, so that the robot's grasping succeeds well. [1, p. 4]

The positionings and grasp detections are achieved using sensors and cameras. The objects to be grabbed are sensed typically as RGB-D (Red Green Blue-Depth) images. These images consist of a typical RGB image and a depth image, which is then transformed into a point cloud image in a 3D space. [3, p. 1679] These images consist of various points in a 3D space, which all have their own x, y, and z coordinates [4]. Using deep learning, these images of objects can then be compared to ones in grasping models. These grasping models are created by either physical interactions or by simulations. The models then help robots to basically memorize different object shapes, so they can grasp similar looking objects by using the same grasping model. [1, p. 3–4]

One of the most important steps in robot grasping is the object's pose estimation. In this step the robot estimates the location and orientation of an object that it recognizes through its database. Pose estimation is required for the robot to assume the grasping position, although motion planning is needed for these to work. An object's 6D position is mainly used in an object's pose estimation, which can be then used in 6-DOF grasping [3, p. 1679]. The acronym 6-DOF means six-degrees-of-freedom. 6D consists of the typical x, y, and z coordinates, which handles the horizontal, vertical, and depth aspects respectively, but also the pitch, yaw, and roll [5]. These last three degrees of freedom are measured in angles, which are analogous to opening a lid (pitch), opening a door (yaw), and rotation (roll) [6]. This 6-DOF is illustrated in figure 2. There are also other types of DOFs that robots use but 6-DOF is the most used one, as it is a requirement to work in a 3D environment.

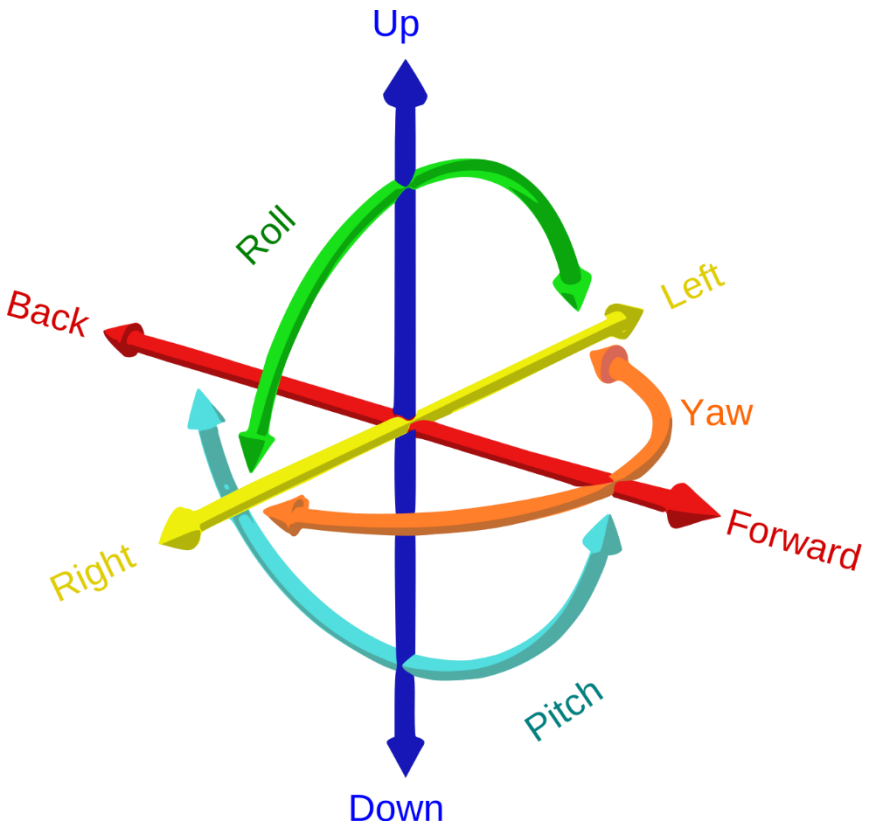



***Figure 2.*** *Illustration of 6D: forward/back is X, left/right is Y, and up/down is Z.* [20]

The gripper part of the robot is also very important when grasping models are created. This is due to the differences between the types of grippers, which can be parallel, three-fingered, or even five-fingered grippers. The gripper part is also known as an end-effector, which includes a wider conception of grippers, for example suction

based end-effectors [3, p. 1678]. The grippers have sensors in them also to detect whether they have grabbed an object, and if the grasping pressure is adequate. The state of the art in robot grasping software is open software that is free to use. Because it is free to use it has no warranty, and no liability in using it. Integrating this software as a part of an enterprise's system can also take time and resources from it. It might also take a long time for an enterprise's developers to learn how to use it. There is also no guaranteed support from the software's original developers, and the software might not even work with some hardware at all. [1, p. 5] The grasping software therefore requires more development, so that everyone can benefit from it.

# 2. THE GRASPING MODELS

In this section six different grasping models are covered briefly, explaining what they are and how they execute grasps. Also, some new concepts will be briefly explained along the way to help the reader understand these models.

## 2.1 6-DOF GraspNet

6-DOF GraspNet is a model that generates diverse grasps for various unknown objects. By its name the model takes advantage of 6-DOF grasping, when determining objects and their locations. Alongside the 6-DOF grasping, the model uses a network of grasp evaluators, that make sure the generated grasps are going to be successful. The network also improves over time in an iterative process. To generate the grasps, the model uses a variational auto-encoder (VAE), which samples a variety of grasps for an object. These samples can be very inaccurate, so that is why the model uses the network of grasp evaluators in unison. [7, p. 8]

The VAE is one of the key aspects of 6-DOF GraspNet. Through it the model can determine many possible grasps for point clouds in various objects. It provides multiple functioning grasps but also includes a few failures as well. After this is done, it gives these grasps to a grasp evaluator, that either accepts or rejects a possible grasp. [7, p. 1–3] The point clouds are captured by using RGB-D cameras. The grasp determinations are estimated by focusing on the center mass of the point cloud in an object. The failed grasps, that are included among the functioning ones, are very close to being successful grasps. Taking this into account, the model can attempt to refine many of these failed grasps into more successful ones. [7, p. 4–5] This process is illustrated in figure 3.

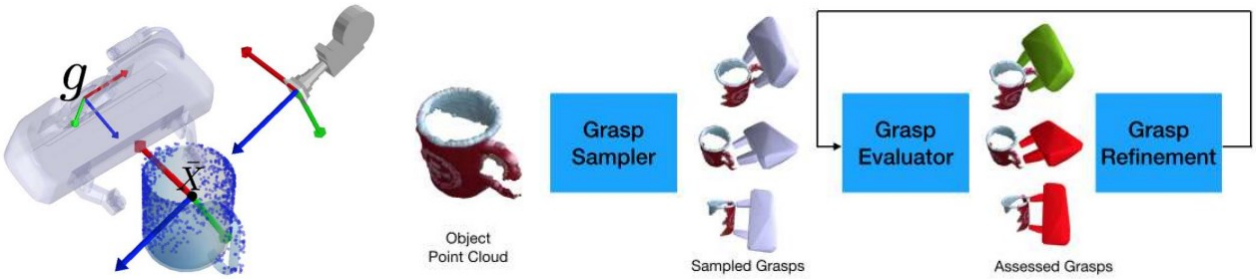



***Figure 3.*** *The VAE process of determining grasps in a point cloud.* [7, p. 4]

In their paper, Mousavian et al. [7] explained how they trained their model. The data used for training is gathered from simulations using FleX, in which grasps for various arbitrary object shapes can be provided realistic simulations [7, p. 5]. In training, the model uses PointNet++ for the grasp generator and the evaluator networks. PointNet++, and PointNet are architectures, that can represent 3D data and extract said representation efficiently [7, p. 3]. Due to the nature of the training, no real data is needed to train the model, only simulations are needed. Also, any kind of end-effector part would do with this model, provided that it has been trained in using the wanted end-effector.

In their paper, Mousavian et al. [7] conducted experiments using their model after providing training for it in simulations. They used their model and a basic 6-DOF grasp planner, and then compared how they performed. They chose a variety of different common household items, and observed which model was more successful. The average success rate for 6-DOF GraspNet was 90%, while the basic 6-DOF grasp planner was 52%, outperforming it vastly. [7, p. 7–8]

## 2.2 Dex-Net 4.0

Dexterity Network 4.0, or just Dex-Net 4.0, is a dataset generation model based on its previous versions Dex-Net 2.0 and 3.0. It is a model that trains a specific set of end-effectors to grasp efficiently in a hybrid approach of synthetic datasets using domain randomization with analytic models of physics and geometry. All grasps that are evaluated by Dex-Net 4.0 take account of task-specific forces and torques, and then try to resist them under random perturbations. Grasp planning is done by using depth images, instead of relying on sensors and manually inputted physical parameters. By taking

depth images, grasps can be effectively and accurately simulated via ray tracing. [8, p. 1]

In their paper, Mahler et al. [8] implemented this model for a robot with two end-effectors: a parallel-jaw gripper and a suction cup-based gripper. They trained their Dex-Net 4.0 model with 1664 unique 3D objects with their dataset containing over 5 million grasps. Both grippers also had their own Grasp Quality Convolutional Neural Network, or CQ-CNN so they could work in unison to plan grasps for each object's point cloud. A CNN is an algorithm utilized in deep learning, that takes an input image and assigns important attributes to it, and then can differentiate various objects from the others [9]. The purpose of this was to show the efficiency of Dex-Net 4.0, with it resulting in very reliable grasping results. According to their experiments, Dex-Net 4.0 managed to achieve 95% reliability on a physical robot. [8, p. 1] Also according to their experiments, using a consistent reward system helps increase the physical robot's reliability when performing grasps across different end-effectors. In addition, regarding their experiments, when the number of objects to be grabbed are increased in a heap, the reliability on those grasps decreases. The same happens when the objects have complex geometries and material properties, such as transparent materials. But in these cases, a memory system can aid to remedy the decreasing reliability from 63% to 80%. [8, p. 5]

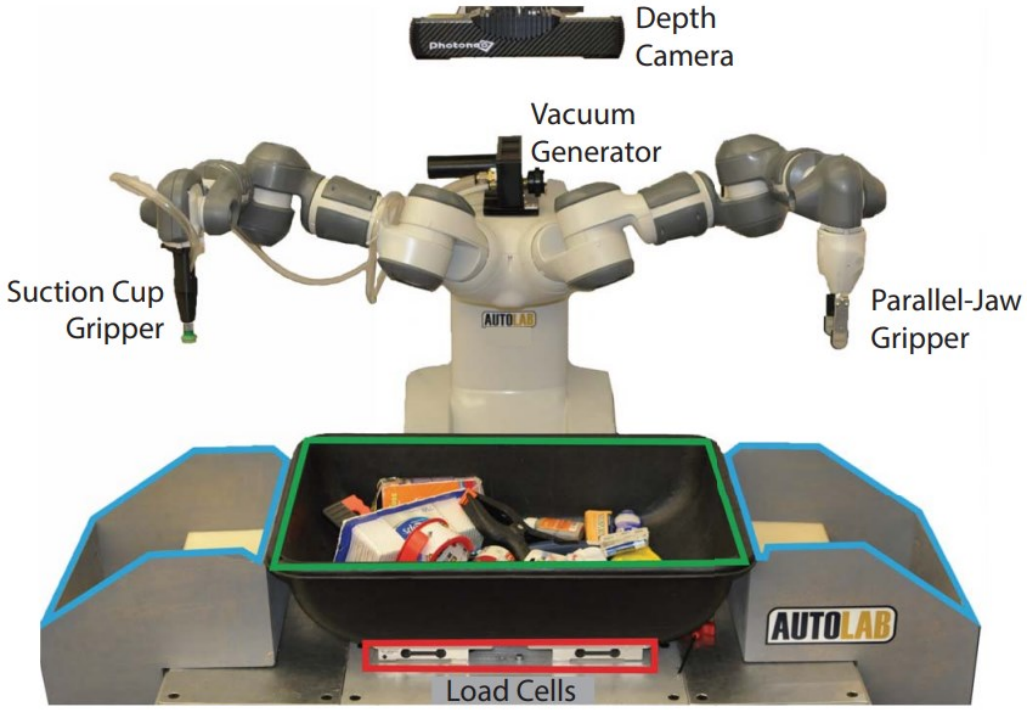



***Figure 4.*** *The setup for Dex-Net 4.0 model's experiment.* [8, p. 3]

Mahler et al. [8] came to these conclusions from the following experiment using the aforementioned setup. The experiment included the robot with the two different end-effectors, two empty blue outlined bins, and one green outlined bin with various objects with different geometries and materials, all of them common household items. This setup is shown in figure 4. The test included more than 2500 grasp attempts and 50 test objects. In the test the robot's depth camera determined whether to use the parallel-jaw gripper, or the suction cup gripper, and then move the objects from the green bin to the blue bins next to it. The objects themselves had two difficulty levels 1 and 2, with the level 2 objects having more complexity. After the test they determined that level 1 and level 2 objects had a grasping reliability of 97% and 95% respectively, and that the robot used the suction cup end-effector on 82% of the grasps. [8, p. 2–4]

## 2.3 GraspNet-1Billion

GraspNet-1Billion is a model that utilizes a multi-object-multi-grasp setting by being a large-scale datanet. It was made for large, cluttered scenarios, that has a rich and dense annotation database for pose estimation. The dataset consists of 88 household objects with high quality 3D models, which have been captured in RGB-D images by two depth cameras in 190 cluttered scenes. Each of these scenes have had 512 images taken by both cameras, bringing in a total of 97,280 images for the whole dataset. All images have 6-DOF annotations for grasp poses, and each scene has a varied amount of grasp poses ranging from 3,000,000 to 9,000,000, bringing in over 1.1 billion grasp poses, from which the model gets its name from. The dataset also has accurate 6D pose annotations, object masks and bounding boxes, and with each frame having an associated camera pose. [10, p. 3] Fang et al. [10] created this model because they proposed that many different researches lack sufficient training data for their robots, and therefore suffer in making them grasp efficiently.

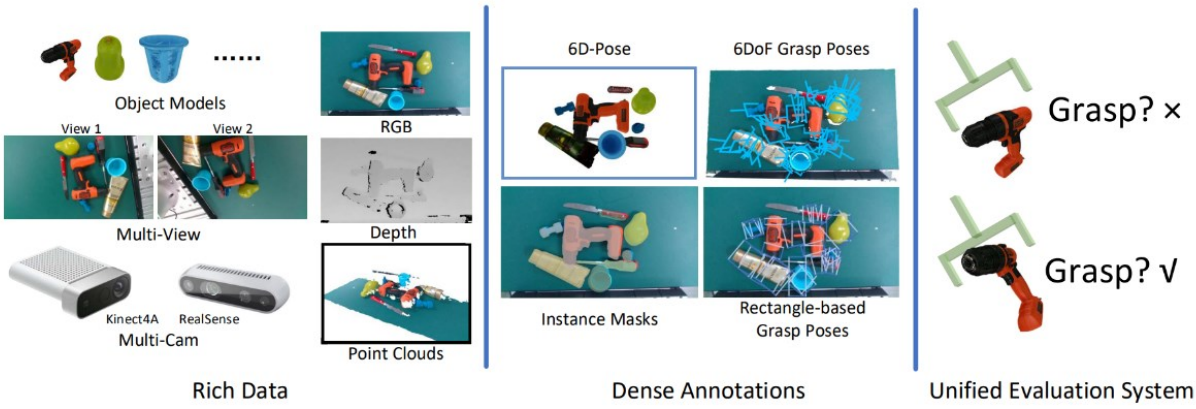



***Figure 5.*** *The model's data gathering and annotation process.* [10, p. 3]

In their paper, Fang et al. [10] described how they collected all the data for their GraspNet-1Billion model. All the objects used in data collection were their own making, from the Dex-Net 2.0, and from the YCB dataset. The latter, Yale-CMU-Berkeley dataset is an object and model set designed for benchmarking robotic manipulation. It has many daily household objects with different shapes, sizes, etc. [11] The data was collected by placing 10 random objects in a heap, and then having a robot with two mounted cameras, as was described before, picking the objects along a fixed movement trajectory. After the pictures were taken, they were annotated with an automated process to avoid a taxing amount of labor. [10, p. 3–4] This process is shown in figure 5.

Fang et al. [10] also detailed how they put together their end-to-end grasp pose detection network. They had to split their data into three different categories for scenes with seen objects, with unseen but similar objects, and novel objects so that data evaluation for grasp pose estimation. When building a solid foundation for viewpoint classification, they used PointNet++ as a base network for capturing point cloud geometry. This way they could get approaching vectors from points in the point clouds that are graspable. They could then use this data to create an algorithm that can create and predict accurate grasp poses in their end-to-end design. [10, p. 5–7]

## 2.4 FFB6D

Full Flow Bidirectional fusion network for 6D, or just FFB6D for short utilizes 6D pose estimation from a single RGB-D image to generate a grasp. The model takes advantage of the taken RGB-D image's data in a two-fold operation, where it analyzes the image from its RGB data, and the depth image data with a point cloud. These two analyses work in tandem in their own separate networks, and fuse together for further analysis at certain intervals, providing a grasp detection at the end. With its RGB based CNN, each pixel of the taken image is used to create a point in XYZ-coordinate map, while the point cloud based PCN (point cloud network) works together with these coordinates to provide points in the point cloud near the coordinates. [12, p. 3–4] These are then used by a 3D key point detection module and an instance semantic segmentation module to provide an object pose estimation. After selecting these key points, a least-squares fitting algorithm is used to generate a grasp pose estimation from said key points for different objects. [12, p. 4–5] This process is illustrated in figure 6.

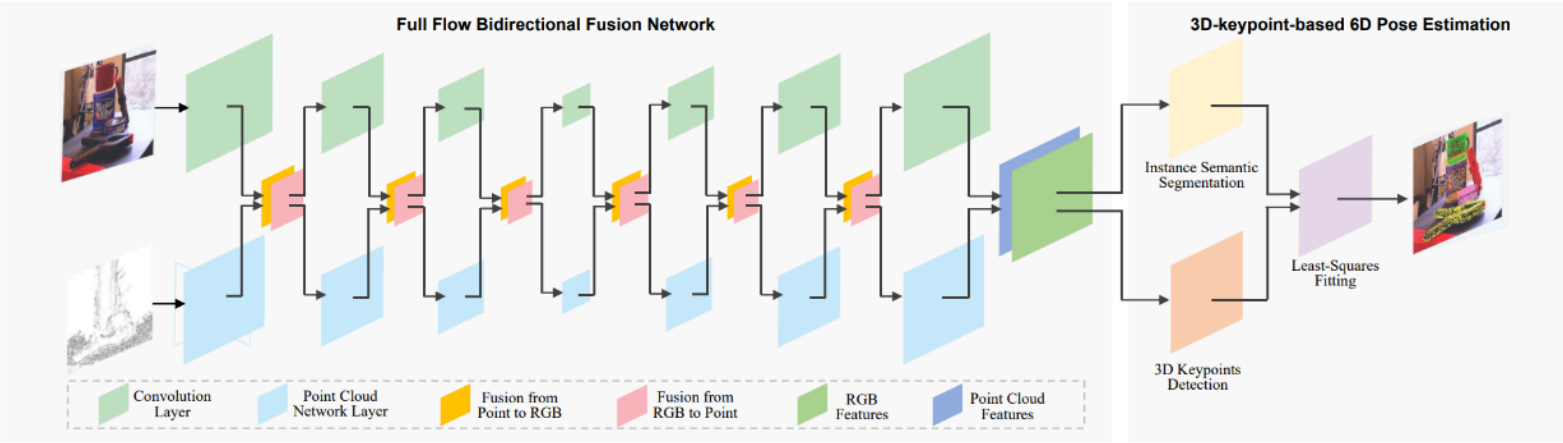



***Figure 6.*** *The model's two networks working in tandem and fusing.* [12, p. 3]

He et al. [12] developed an earlier grasping model called Point-wise Voting Network 3D, or just PVN3D for short, which they used in their new FFB6D model when determining 3D key points. PVN3D is a 6-DOF based grasping model, that utilizes 3D key points to train a robot to effectively grasp an object. Key points are a couple of points in an object's point cloud, that the model's algorithm uses to accurately determine the shape of the object. As with many other models, RGB-D pictures are utilized to get these point clouds in the first place. He et al. [13] stated in their paper that current key

point-based models at first detect 2D key points, so that they can be utilized to estimate 6D poses with a PnP algorithm [13, p. 2]. PnP stands for Perspective-n-Point, which is defined as a problem of finding the relative pose between an object and a camera. This pose is determined from a set of “n” pairing between 3D points of an object, and their corresponding 2D projections on a focal plane. [14] He et al. [13] stated that 2D key point-based methods aim to minimize 2D projection errors of objects, but that in the real 3D world these small errors may be large. However, RGB-D sensors have cheapened, which allowed them to do everything fully in 3D by using depth images. [13, p. 2] They also concluded that 3D key point-based approach in these models is a promising way to tackle the 6-DOF pose estimation problems [13, p. 8].

PVN3D is based on a deep 3D Hough voting network, that uses a multi-tasking algorithm with three different modules: a key point detection module, a semantic segmentation module, and a center voting module. These modules are for predicting key point offsets per-point, predicting semantic labels per-point, and predicting offsets to an object’s center per-point respectively. [13, p. 3] Hough voting itself is a computer vision and image processing technique, that can be used to detect patterns in an image, which are then represented by mathematical curves or shapes [15]. With the algorithm using these three modules working in tandem, they boost each other’s performances and increasing the learning efficiency. After the multi-tasking is complete, there is one more algorithm in use for 6-DOF pose estimation that uses least-square fitting to achieve its goal. [13, p. 3–4]

He et al. [12] explained how they trained their model by using three different datasets: the YCB-Video dataset, LineMOD dataset, and Occlusion LineMOD dataset. LineMOD is a dataset that consists of 13 low-textured objects in 13 videos with varying lighting and cluttering. Occlusion LineMOD is a modified dataset from the original LineMOD, in which multi annotated objects are heavily occluded. [12, p. 5] According to their paper, He et al. [12] didn’t perform any physical experiments, training their model entirely with simulations. After performing their experiments, their model managed to perform over 90% successful grasp poses with the LineMOD and YCB-Video datasets, and over 65% successful grasp poses with the Occlusion LineMOD dataset. They concluded that their model outperformed various other models in these same experiments, and that their model’s implementation of a full flow bidirectional fusion network has more applications, and that more research should be done to explore this further. [12, p. 7]

## 2.5 ROI-GD

ROI-GD is a Region of Interest based grasp detection model, in which it bases all grasps on regions of taken images that are likely to contain objects to be grabbed. It creates grasps in a two-part process. In the first part, RGB images are taken so that the ROIs can be generated. In the second part, grasp detections are made based on the generated ROIs. When generating regions of interest, the taken RGB image is split into different parts using CNN to extract deep features of the image. Then a Region Proposal Network, or RPN for short, is made from 3 x 3 convolutional layers to provide bounding boxes for objects in order to generate grasp detections. These bounding boxes are in fact the regions of interest. After these ROIs are generated, they can then be used for grasp detections one ROI at a time, and then be combined into a singular image with all the grasp detections present. This architecture is presented on figure 7. Zhang et al. [16] explained in their paper, that many other models try to create grasp detections purely on RGB or RGB-D images, treating the entire images as singular ROIs, and that their model separates the input image into multiple ROIs. This works well with multiple objects in a scene, where the objects are overlapping each other. [16, p. 2–3]

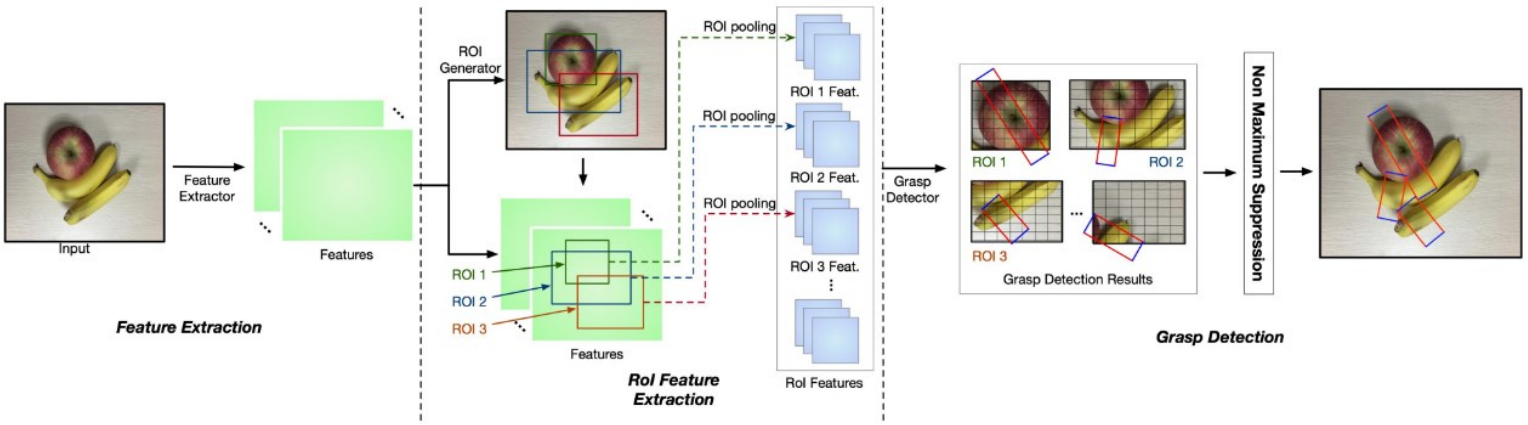
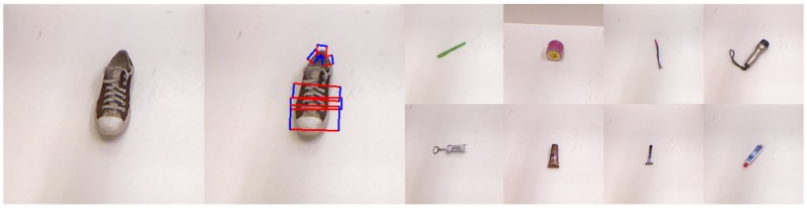



***Figure 7.*** *The model's architecture and process visualized.* [16, p. 3]

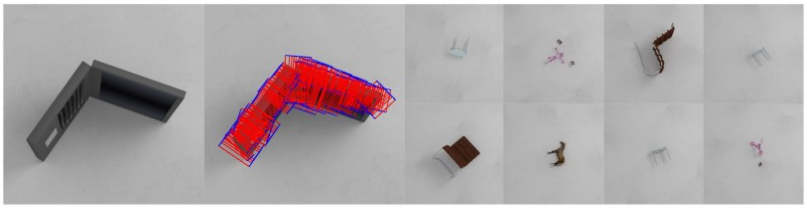
In their paper, Zhang et al. [16] explained their ROI method further by saying that they made a new network in ROI-GD to create grasp detections for multiple objects in heaps. While their model makes the bounding box ROIs as was explained earlier, it also detects the specific objects and annotates them. With both ROIs and the object details working in tandem, it results in a more accurate grasp detection for each object. [16, p. 3–4] Based on experiments made with Cornell Grasp Dataset, Jacquard

Dataset, and Visual Manipulation Relationship Dataset, their model compares with other state-of-the-art grasp detection models in single-object grasps. However, in multi-object grasps their model seems to perform better. [16, p. 8]

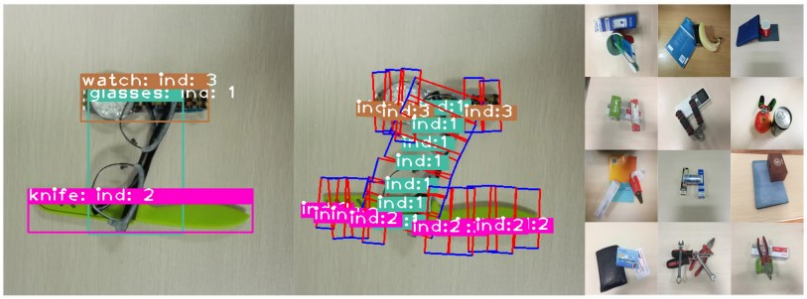
Zhang et al. [16] trained their model to make multi-object grasps by using the Visual Manipulation Relationship Dataset, or VMRD for short includes 5185 images with over 17000 object instances and 51000 manipulation relationships. With single-object grasps they used Cornell Grasp Dataset and Jacquard Dataset, which contain 885 images and 54000 images respectively. [16, p. 5] Examples of how these datasets are distinguished are shown in figure 8. In the training and the experiments, they used a robot with two arms utilizing 7-DOF instead of 6-DOF [16, p. 7]. 7-DOF contains all the degrees of freedom that 6-DOF has, however it comes with the ability to move the robot from one place to another along an axis [17]. The robot in training had identical end-effectors, with them being two parallel fingered grippers. From their experiments they concluded that in single object scenes their model can achieve grasps on a 97.5% and 92.5% success rate for prediction and execution respectively. On multi-object scenes the model had a success rate of 88.8% and 83.8% for prediction and execution respectively. [16, p. 7]



(a) Examples of Cornell Dataset



(b) Examples of Jacquard Dataset




(c) Examples of VMRD

***Figure 8.*** *Example images in the three datasets used in the model.* [16, p. 5]

## 2.6 GR-ConvNet

Generative Residual Convolutional Neural Network, or GR-ConvNet for short is a grasping model that generates grasp detections by taking RGB-D images, and then creates three different pictures to generate these grasps. The images are gone over pixel by pixel in the model, that includes quality, angle, and width. After these three have been analyzed by the model, it then generates a grasp based on the data gathered from the three generated images. It has two modules that handle this model, one of which is the inference module that goes over the aforementioned parts, and the other being the control module, which handles the motion of the robot, such as trajectory planning and task controlling. [18, p. 1] These modules are represented in figure 9.

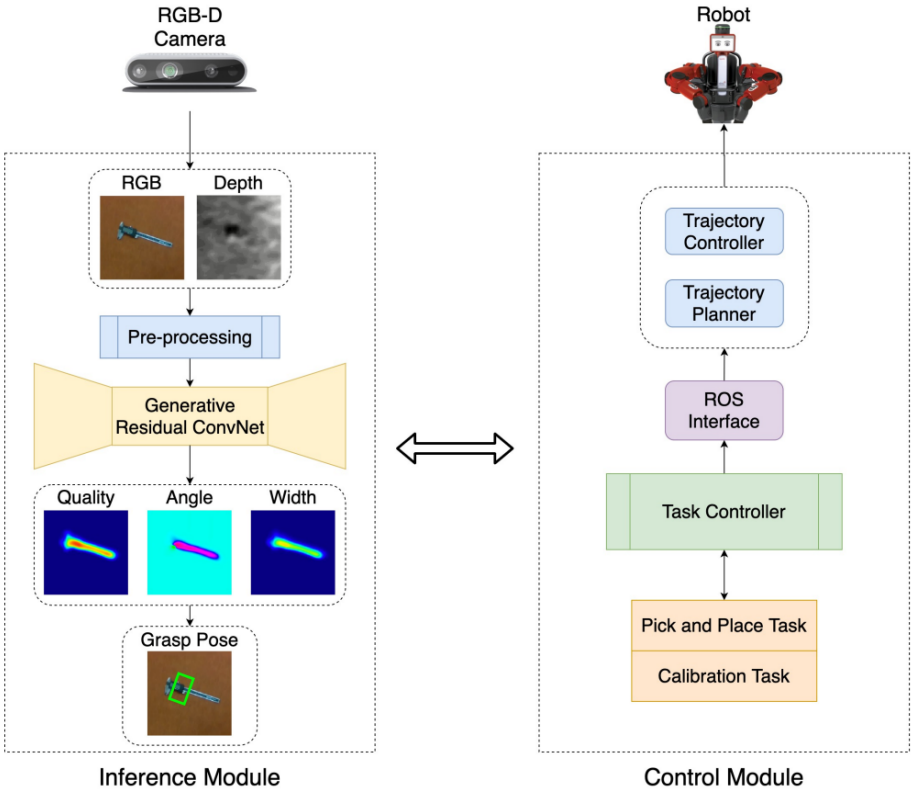



***Figure 9.*** *The model's two modules: the interference and control.* [18, p. 1]

In their paper, Kumra et al. [18] went into further detail about their model and how it extracts the data from the depth image so that it can be split into three images concerning quality, angle, and width, although technically the angle image actually consists of two images regarding sine and cosine combined into one. The image goes through many different convolutional layers in the algorithm to extract data, so they can

be reconstructed into the three desired images that can then be turned into grasp detections. There are three convolutional layers that the input image goes through, which then passes through five residual layers, and then three other convolutional transpose layers to generate the images. This generative residual convolutional network is illustrated in figure 10. Their model focuses on treating the input image heavily so they can generate reliable grasp detections. [18, p. 3–4]

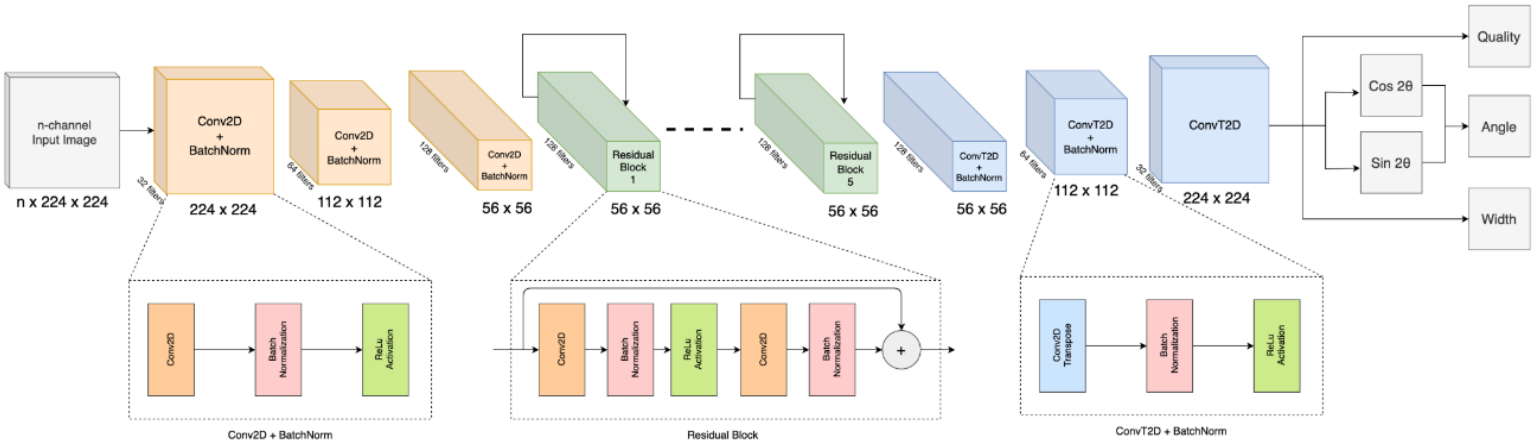



***Figure 10.*** *The proposed generative residual convolutional network's architecture.* [18, p. 4]

For training their model Kumra et al. [18] used the Cornell Grasp Dataset and Jacquard Dataset. Unlike previously mentioned usage of Cornell Grasp Dataset, in this model an extended version of Cornell Grasp Dataset was used, which contained 1035 images of objects and grasps instead of 885 images. Objects used in training their model included household objects, more complex adversarial objects, and objects in heaps. A 7-DOF robot with a two-fingered parallel end-effector was used to execute grasps. Grasps over all these datasets and experiments yielded over 93% success rate, with failures being mostly objects that were already grabbed by the robot but slipped away from the end-effector for various reasons, such as bumping into another object in a heap. [18, p. 5–7]

# 3. COMPARISONS

In this section the previously covered six grasping models are compared with each other. Attributes gathered and compared in the following two tables are: method of grasping, means of acquiring data for grasping, datasets a model uses, training a model, end-effector used for grasping, and experiments done to test a model.

| | **6-DOF GraspNet [7]** | **Dex-Net 4.0 [8]** | **GraspNet 1Billion [10]** |
|---|---|---|---|
| *Method* | Uses 6-DOF grasping with a VAE to determine multiple grasps in a point cloud with a network of grasp evaluators. | Trains specific sets of grippers to grasp efficiently by domain randomization with analytic models of physics and geometry. | Utilizes a multi-object-multi-grasp setting by being a large-scale datanet, mostly for large, cluttered scenarios; it has a rich and dense annotation database for pose estimation. |
| *Data modality* | RGB-D camera focuses on the center mass of a point cloud. | Depth images are taken and evaluated in simulations with ray tracing. | Two RGB-D cameras taking pictures of various cluttered scenes, which are annotated. |
| *Dataset(s)* | Custom dataset simulated with FleX; boxes, cylinders, bowls, bottles, mugs. | Custom dataset generated by the model itself; 5000 unique object heaps. | Custom dataset made for the model alongside Dex-Net 2.0 and YCB datasets. |
| *Training* | Various simulations only with PointNet++ used for grasp evaluations. | Using various 3D objects in simulations to generate 5 million grasps. | Simulations through 100 different scenes with refining different grasps by separating them by score. |

| | | | |
|---|---|---|---|
| *Gripper used* | Simulations used a parallel fingered gripper; however, any type of gripper should be fine. | Model used parallel fingered gripper, and a suction cup gripper. | Two-fingered parallel gripper was the focus of this model. |
| *Experiments done* | Parallel finger gripper 7-DOF robot performed grasps on items similar to the ones in simulations. | A robot armed with two gripers; parallel fingered and a suction cup were used to move objects to two different bins. | Only simulations done with PyTorch, Adam optimizer and an Nvidia graphics driver; with the parallel fingered gripper in mind. |

***Table 1.*** *Six grasping models previously covered, broken down; the first half.*

| | **FFB6D [12]** | **ROI-GD [16]** | **GR-ConvNet [18]** |
|---|---|---|---|
| *Method* | Utilizes 6D pose estimation from a single RGB-D image; breaks image down in a two-fold operation. Detects key points in a point cloud for grasp detection. | Uses Regions of Interest as its basis; parts of a scene with most likely to have objects for grasping. Uses different CNNs for analysis, combining split parts into a whole with grasp detection. | Takes RGB-D images and splits them into three separate images, which the inference module handles. Afterwards the control module executes the grasp based on the inference module's data. |
| *Data modality* | RGB-D picture is taken and divided into two categories for analysis: RGB data and a depth image with a point cloud. | RGB image is taken and split into different parts for analysis: each object has its own split image based on regions of interest. | RGB-D images are split into three separate images concerning angle, quality, and width; angle image is technically two images combined into one. Data is extracted from these in convolutional layers. |

| | | | |
|---|---|---|---|
| *Dataset(s)* | Uses three different datesets: YCB-Video dataset, LineMOD dataset, and Occlusion LineMOD dataset. | For multi-object grasps the model uses VMRD, and for single object grasps: Cornell Grasp Dataset and Jacquard Dataset. | Two different datasets are used by this model: Jacquard Dataset, and an extended version of Cornell Grasp Dataset. |
| *Training* | Using the three different datasets, simulations were made to train the model. | Each dataset is handled by the model; annotating objects in the datasets heavily, and by using bounding boxes to separate objects in clutters. | Each image from the two different datasets were ran through the model's process. |
| *Gripper used* | No specified gripper was mentioned; model focused mostly on object detection. | Model used two-fingered parallel grippers. In theory any sort of gripper would do depending on objects. | Model used two-fingered parallel grippers. The model's developers have stated that implementation of different gripper types must be researched. |
| *Experiments done* | No physical experiments were made, only simulations to train the model. | A 7-DOF robot with two arms, both armed with two-fingered parallel grippers performed grasps on multi-object and single-object scenes. | A 7-DOF robot that was armed with a two-fingered parallel gripper was used to grasp objects. |

***Table 2.*** *Six grasping models previously covered, broken down; the second half.*

There are many similarities in some aspects of these six models, and quite a few differences as well. When it comes to the methods of these models, they are indeed different from each other. Some similarities occur when it comes to them, such as 6-DOF GraspNet's and FFB6D's methods using 6D grasping and heavily relying on point cloud analysis. When it boils down to data acquisition, RGB-D sensors seem to be almost universal. How the data is processed after these types of sensors take images

varies somewhat on the method of the grasping model. The images are either used in simulations or processed according to the method of the grasping model. Datasets always vary depending on the model, with some of them making their own datasets. However, there are some popular datasets used by various models, such as the Cornell Grasp Dataset, and the Jacquard Dataset.

The models covered here mostly trained with different simulations in a virtual 3D environment, except for ROI-GD and GR-ConvNet. The latter have their own unique training methods, which are based on the grasping model's data analysis process. When it comes to end-effectors, the parallel fingered gripper with two fingers is the most popular one. Some of these models seem to be purely based on said end-effector, however a couple of models should work with multiple types of grippers, such as the 6-DOF GraspNet. All the models had physical experiments done with the model, usually done with a 7-DOF robot arm, all except for GraspNet 1Billion and FFB6D. The latter only performed simulations for their model to train it, and the former performed simulations in a virtual 3D environment to grasp objects in their dataset.

# 4. CONCLUSION

In this paper, six different models for robot grasping were covered: 6-DOF GraspNet, Dex-Net 4.0, GraspNet 1Billion, FFB6D, ROI-GD, and GR-ConvNet. Robot grasping itself and its deep learning were also covered briefly to give the reader context how these grasping models operated. To sum the process of grasping, it goes as follows: grasp detection, object pose estimation, motion planning, grasping positioning, and finally the implemented grasp. Deep learning is the process, in which a grasping model teaches itself to grasp objects in an end-to-end manner, in other words, without third party help. RGB-D cameras are a necessity to achieve very reliable grasps, and 6-DOF ensures that a robot can work in 3D environment.

The grasping models had some similarities when they were compared with each other, however their overall method was unique to their own respective models. 6-DOF GraspNet uses 6-DOF grasping with a VAE to determine multiple grasps in a point cloud with a network of grasp evaluators. Dex-Net 4.0 trains specific sets of grippers to grasp efficiently by domain randomization with analytic models of physics and geometry. GraspNet 1Billion utilizes a multi-object-multi-grasp setting by being a large-scale datanet, mostly for large, cluttered scenarios. FFB6D utilizes 6D pose estimation from a single RGB-D image; breaks image down in a two-fold operation; it detects key points in a point cloud for grasp detection. ROI-GD uses Regions of Interest as its basis; parts of a scene with most likely to have objects for grasping. GR-ConvNet takes RGB-D images and splits them into three separate images, which the inference module handles, and afterwards the control module executes the grasp based on the inference module's data.

In conclusion none of these grasping models would have worked without an RGB-D camera. The depth images provide highly accurate and useful data for the models' analysis processes. However, in the future there may be even more accurate ways to gather data with than an RGB-D camera but right now it seems to be the best option available. When conducting experiments with grasping models, they either boil down to simulations and/or physical experiments with a robot, armed with an adequate end-effector.

# SOURCES


[1] A. M. Sefat, S. Ahmad, A. Angleraud, E. Rahtu, R. Pieters, Deep Learning for Robot Perception and Cognition.

[2] H. A. Pierson, M. S. Gashler, Deep learning in robotics: a review of recent research, Advanced Robotics 31 (16) 2017 pp. 821–835.

[3] G. Du, K. Wang, S. Lian, Vision-based robotic grasping from object localization, pose estimation, grasp detection to motion planning: A review, arXiv preprint arXiv:1905.06658, Springer Nature 2019.

[4] Point Cloud. ScienceDirect. Multimaterial 3D Printing Technology. Referred to (23.10.2021). Available at: https://www.sciencedirect.com/topics/engineering/point-cloud

[5] 6DOF. PCMag. Referred to (23.10.2021). Available at: https://www.pcmag.com/encyclopedia/term/6dof

[6] Pitch-yaw-roll. PCMag. Referred to (23.10.2021). Available at: https://www.pcmag.com/encyclopedia/term/pitch-yaw-roll

[7] A. Mousavian, C. Eppner, D. Fox, 6-DOF GraspNet: Variational Grasp Generation for Object Manipulation. Available at: https://openaccess.thecvf.com/content_ICCV_2019/papers/Mousavian_6-DOF_GraspNet_Variational_Grasp_Generation_for_Object_Manipulation_ICCV_2019_paper.pdf

[8] J. Mahler, M. Mahl, V. Satish, M. Danielczuk, B. DeRose, S. McKinley, K. Goldberg, Learning ambidextrous robot grasping policies. Available at: https://goldberg.berkeley.edu/pubs/Ambidextrous-Grasping-Science-Robotics-Jan-2019.pdf

[9] A Comprehensive Guide to Convolutional Neural Networks – the ELI5 way. SaturnCloud. Referred to (8.8.2023). Available at: https://saturncloud.io/blog/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way/

[10] HS. Fang, C. Wang, M. Gou, C. Lu, GraspNet-1Billion: A Large-Scale Benchmark for General Object Grasping. Available at: https://openaccess.thecvf.com/content_CVPR_2020/papers/Fang_GraspNet-1Billion_A_Large-Scale_Benchmark_for_General_Object_Grasping_CVPR_2020_paper.pdf

[11] YCB Benchmarks – Object and Model Set. Referred to (13.8.2023). Available at: https://www.ycbbenchmarks.com

[12] Y. He, H. Huang, H. Fan, Q. Chen, J. Sun, FFB6D: A Full Flow Bidirectional Fusion Network for 6D Pose Estimation. Available at: https://openaccess.thecvf.com/content/CVPR2021/papers/He_FFB6D_A_Full_Flow_Bidirectional_Fusion_Network_for_6D_Pose_CVPR_2021_paper.pdf

[13] Y. He, W. Sun, H. Huang, J. Liu, H. Fan, J. Sun, PVN3D: A Deep Point-wise 3D Keypoints Voting Network for 6DoF Pose Estimation. Available at: https://openaccess.thecvf.com/content_CVPR_2020/papers/He_PVN3D_A_Deep_Point-Wise_3D_Keypoints_Voting_Network_for_6DoF_CVPR_2020_paper.pdf

[14] An Event-Based Solution to the Perspective-n-Point Problem. Frontiers. Referred to (16.8.2023). Available at: https://www.frontiersin.org/articles/10.3389/fnins.2016.00208/full#:~:text=The%20goal%20of%20the%20Perspective,projections%20on%20the%20focal%20plane.

[15] Hough Voting. Medium. Referred to (18.8.2023). Available at: https://medium.com/@sim30217/hough-voting-is-a-technique-used-in-computer-vision-and-image-processing-to-detect-patterns-in-an-f288e0ef6605#:~:text=Hough%20voting%20is%20a%20technique%20used%20in%20computer%20vision%20and,types%20of%20shapes%20as%20well.

[16] H. Zhang, X. Lan, S. Bai, X. Zhou, Z. Tian, N. Zheng, ROI-based Robotic Grasp Detection for Object Overlapping Scenes. Available at: https://arxiv.org/pdf/1808.10313.pdf

[17] Unraveling Degrees of Freedom and Robot Axis: What does it mean to have a multiple axis pick and place or multiple axis robot? Motion Control Robotics. Referred to (10.9.2023). Available at: https://motioncontrolsrobotics.com/unraveling-degrees-of-freedom-and-robot-axis-what-does-it-mean-to-have-a-multiple-axis-pick-and-place-or-multiple-axis-robot/

[18] S. Kumra, S. Joshi, F. Sahin, Antipodal Robotic Grasping using Generative Residual Convolutional Neural Network. Available at: https://arxiv.org/pdf/1909.04810.pdf

[19] IRB 4400 Industrial Robot. The heart of Robotics. Available at: https://library.e.abb.com/public/f8b8c4721404116cc1257b130056d106/IRB4400_R1_US%2002_05.pdf

[20] The six degrees of freedom: forward/back, up/down, left/right, yaw, pitch, roll. Wikimedia Commons. Referred to (26.11.2023). Available at: https://en.wikipedia.org/wiki/Six_degrees_of_freedom#/media/File:6DOF.svg