Machine Learning for Robotic Grasping

Jack Rasiel

Advisor: Kostas Daniilidis (University of Pennsylvania)
## Contents:

**Overview**………………………………………………………………………………………………1

**Part 1:** Literature Review………………………………………………………………………………2  
*(A Review of Supervised Learning-based Approaches to Grasp Synthesis)*

**Part 2:** Research Contribution…………………………………………………………………………20  
*(Learning “Sensible” Grasps  
With a Novel Application of Object Pose Detection and Synthetic Training Data)*

**Part 3:** Annotated References…………………………………………………………………………35

## Abstract:

This thesis is comprised of two primary sections: a literature review and a research contribution. In the literature review, I first introduce the problem of robotic manipulation and grasping, and give context for why it is an important area of research in robotics. I then define the problem of vision-based robotic grasp synthesis, and survey recent research, which has focused on supervised learning-based solutions. I identify a common structure to learning-based systems, and a set of distinct sub-problems every system must address (e.g., how to represent grasps in sensor data, and how to collect data for training). With this perspective, I then identify the outstanding problems in the field, and make several recommendations for future research.

The second part of the thesis, my contribution, attempts to address several of those outstanding problems. Notably, my approach to grasp synthesis is based on CNN-based object pose detection, which is unique among the literature reviewed. My primary goal is to choose grasps based on object-level properties, rather than simply local texture cues. I also develop new tools and methods to create training data for this system, and demonstrate their usage in a proof-of-concept system. I discuss the challenges faced in implementation and testing, and identify practical steps to improve it in the future. I also review recent developments in object and human pose detection research, and identify how they could be utilized to create better grasping systems. Lastly, I discuss the broader impact that object pose detection could have on grasping systems, and recommend research directions to further this.
**Part 1: A Review of Supervised Learning-based Approaches to Grasp Synthesis**

1 – Introduction & Motivations

1.a: Putting Grasp Synthesis in Context.

Robots, by their definition, interact with the physical world. The subject of *robotic manipulation* encompasses the infinite manifestations of these interactions. As is so often the case in robotics, manipulation is a diverse and interdisciplinary subject, spanning mechanical engineering, electrical engineering, artificial intelligence, control, etc. Examples include wheeled robots which push obstacles out of their way to navigate a cluttered environment, surgical robots which replicate the movements of their operator at a thousandth the scale, and humanoid robots which can open heavy doors and operate power tools to aid in disaster recovery.

*Grasping* is an especially important and compelling topic in manipulation. As the primary mode of manipulation for humans, our built world is designed with grasping in mind. Thus, if robots are to perform useful work in human spaces, they must be able to grasp and manipulate objects reliably.

Like manipulation in general, grasping includes many distinct sub-problems which can be studied through different disciplinary lenses. Mechanical engineering can provide novel manipulator designs and the analytical tools needed to reason about their performance; electrical engineering can develop new sensors for tactile and force feedback; and computer science can design algorithms for perception, planning, and control. This paper is concerned with *perception for grasping*—that is, using data from the robot's sensors to decide how best to grasp an object. But even this is still too loosely defined: for example, the quality of a grasp is contingent on the requirements of the task to be performed. I further specify *grasp synthesis* for *pick-and-place* grasping: given an object, choose a configuration of the robot's manipulator which best allows it to
pick the object up, hold it without dropping it, and then put it back down.

Pick-and-place grasp synthesis may seem like a trivial problem, but it has both immediate applications and broader intellectual value. In the short-term, robust pick-and-place grasping could see applications to inventory management in large warehouses and retail storefronts (see, the Amazon Picking Challenge [1]). But more broadly, this problem is a useful distillation of some of the fundamental challenges in manipulation, and solving them in this context will undoubtedly yield insights into more complex problems.

1.b: Machine Learning for Grasp Synthesis.

The real world is complex, varied, and unpredictable. In the context of grasping, objects vary considerably in their geometry, material properties (e.g. compliance and surface friction), and mass distribution. Given this complex parameter space, it is infeasible to manually determine a set of heuristics for choosing grasps with satisfactory reliability. Indeed, in the past ten years applications of machine learning, especially under the paradigm of supervised learning, have raised the bar for grasping performance\(^1\). Where previous approaches to grasp synthesis were unreliable even in constrained environments, learning-based systems perform well in settings better approximating the messiness that exists outside the lab. While these systems are still too unreliable for real-world use, this work is in its early days, and undoubtedly the first systems which do cross that threshold will be part of their legacy.

2 – Background

2.a: Overview of High-level Methodologies in Grasp Synthesis

The learning-based paradigm explored here emerged from a larger pre-existing body of research on grasp synthesis. To best understand the learning-based approaches, it is necessary to

---

\(^1\) It is important to note, also, recent advances in reinforcement learning techniques. While this work is very promising, it is also quite distinct in its objectives and methods from supervised learning-based approaches. Of course, there are overlaps between them, and we will bring up RL-based papers when they offer insights into the topic at hand. However we consider a systematic review to be outside the scope of this paper.
have a grounding in some important ideas from the broader subject.\footnote{2} Bohg et. al. \cite{Bohg2018} judges the highest-level distinction in grasp synthesis research to be between analytic and empirical methodologies. Analytic methodologies construct a mathematical model for grasping, define a set of properties for grasps, and then derive from those properties a set of higher-level “quality metrics”. Then the problem of grasp synthesis becomes a constrained optimization over those quality metrics. In contrast, empirical (“data-driven”) methodologies sample candidates from the space of possible grasps and rank those candidates according to a metric. This metric is defined based on some manner of previous grasping experience, and can be realized in many ways (e.g. a statistical model, a grasp simulator, a database of previous grasps, etc.).


Several basic concepts from analytical grasping are employed frequently throughout the grasping literature, including in the work reviewed here. I now give a brief overview of them (for a more rigorous overview, see \cite{Nagata2017}).

In the analytical context, grasping is defined in terms of wrenches\footnote{3} imparted upon an object by effectors (e.g. the fingers of a dexterous hand) at discrete contact points. A grasp is said
to be in *equilibrium* if the contacts can exert a net wrench of zero magnitude upon the object: for example, by counteracting its weight, and any other external forces acted upon it ([4]).

A pair of related but distinct concepts are *form closure* and *force closure*. If a grasp exhibits *form closure*, then any movement of the object (relative to the contacts) will result in a collision with one of more contacts; thus the object is totally immobilized by the grasp. *Force closure*, meanwhile, describes a condition where contacts are able to impart an arbitrary wrench on the object. Note that force closure does not necessarily imply form closure (see fig. 1).

Dexterous multi-fingered hands can achieve force-closure grasps called *power grasps* by forming a tight cage around the object; parallel jaw grippers cannot cage, and instead execute *clamping grasps*, which are modeled as two parallel planes pressing in towards the object. A clamping grasp may exhibit form closure, but in most cases cannot achieve this, and must instead aim for force closure through applying pressure and inducing frictional forces (see fig. 2).

Lastly, we introduce the *grasp wrench space* and the *ε-metric*, concepts which provide insight into the closure properties of a grasp. The grasp wrench space (GWS) is the space of wrenches that can be applied to an object by a grasp's contact points without causing the object to move. This is calculated as a 6D (three dimensions each for force and torque) convex hull. If a grasp's GWS contains the origin, then that grasp has form closure ([5]). The ε-metric quantifies the “worst case” wrench for a grasp: the magnitude of the minimum wrench which can cause the object to slip ([6]). It is defined geometrically as the radius of the largest sphere that fits within the grasp wrench space; that is, the minimum distance from the origin to the boundary of the GWS hull. The ε-metric is one of the most frequently used quality metrics in grasping simulation.

2.c: Rationale for Data-Driven Grasping.

Data-driven grasping is also known as “empirical grasping”, and this reflects its methodology. Rather than creating a formal model of the grasping problem, data-driven approaches judge the quality of new grasps based on some representation of previous grasping experience. This
category includes the machine learning-based approaches reviewed in this paper.

One rationale for data-driven grasping is the limited applicability of analytical approaches to practical, real-world implementation. Analytical approaches must make simplifying assumptions about some physical and/or dynamical properties, to prevent the problem from becoming intractably complex. For example, object-affecter interactions are often modeled solely through rigid-body dynamics, which fall short in describing the graspability of non-rigid objects (e.g., those made of flexible cloth). Additionally, analytical approaches are typically not robust to the errors in sensing and control inherent in robotics: an attempt to execute a highly-rated grasp may fail if the robot's arm accidentally drifts into a poorly-rated neighbor. Studies which systematically replicated analytically-synthesized grasps on real robots have validated many of these concerns, showing limited correlation between predicted and actual success (\cite{2}) .

Additionally, analytical methods require knowledge of the target object's geometry and physical properties. This limits their applicability in situations where this data is unavailable, such as when the robot lacks 3D sensing capabilities, or when the application requires grasping
unfamiliar objects (i.e., ones for which CAD models are not available). For a comprehensive review of data-driven grasping, we recommend Bohg et. al.'s survey ([2]).

3 – Overview of Topics in Learning-based Grasp Synthesis

Current learning-based grasp synthesis systems share the same basic structure, shown in fig. 3. Given an image (or point cloud, etc.) of a scene containing the object to be grasped, we define a space of possible candidate grasps. I also define a representation of those grasps in terms of the input image. I then sample candidate grasps from that space, and extract the corresponding representations from the input image. These representations are then passed to a model which has been trained (on a dataset of annotated example grasp candidates) to predict the success of a given grasp. An “optimum” grasp is chosen based on the model's outputs for all candidates. If the system is being implemented on an actual robot, the grasp can then be executed by means of standard planning and control algorithms.

There is significant variability in the current literature as to how each of the aforementioned subsystems is implemented. And indeed, as we will see, some of the most compelling work makes the greatest changes to this framework. I will organize my overview of the literature into individual subjects, corresponding roughly (though not exactly) to the subsystems outlined in the previous paragraph.

I begin by examining the various approaches to framing the grasp synthesis problem: as one of classification, of ranking, regression, etc. This decision informs how the rest of the system is implemented, from the structure of training data, to the choice/design of machine learning algorithm, to procedures for training and testing. I then examine the usage of different learning algorithms, and discuss their relative benefits and drawbacks.

Next, we discuss approaches to grasp representation. By this we mean the methods used to extract a useful representation of specific grasps from sensor data. This problem is twofold. First, the representation must structure the sensor data in a way that conveys a specific grasp. For
example, we may map a rectangle in an image to a grasp made by a parallel jaw gripper, orienting the gripper parallel to the object's surface normal at the rectangle's center (see [8]). Second, the sensor data must be presented in a format which facilitates learning. This entails post-processing to both extract useful features and simplify the input parameter space. In the image rectangle representation, we could calculate the texture energy, surface normals, etc. These decisions have significant impacts on the capabilities of the system.

Once the representation of grasps is established, we must then consider how to acquire large datasets of these grasps for training. Acquiring adequate training data is a fundamental concern for any supervised learning problem, but for grasp synthesis (and robotics in general) it is especially challenging, due to the cost and labor of running and supervising trials on real robots. I distinguish approaches to dataset generation based on how grasps are selected and evaluated. Each decision has the potential to bottleneck the size and quality of a grasp dataset. For example, early papers which used manual labeling typically had datasets on the order of 100 to 1000 grasps. More recent papers have increased dataset sizes by an order of magnitude by removing the human component in favor of simulation or robotic self-supervision.

I dedicate extra consideration to simulation-based dataset generation. Simulation avoids some of the problems of both manual labeling (laboriousness, dubious quality) and robot trials (costliness, the need for large amounts of robot time). However, it has problems all its own, stemming from the shortcomings of analytical metrics and the differences in statistics between real and synthetic images. I review these problems in depth, along with attempts to address them in recent work.

Lastly, we address the testing and evaluation of grasp synthesis systems. There are a variety of testing methodologies employed in the literature, designed to evaluate various aspects of a grasp synthesis system’s performance. For example, tests may emphasize the system’s robustness to cluttered scenes, or its performance on objects from outside its training set. Identical trials may
be performed on systems trained with different input data modalities (e.g., RGBD vs. RGB vs. depth-only), to give insight into which are the most useful. I review the results of tests (among others), and discuss their implications. I also note various problems which arise only when testing the system in the real world. For example, the potential positioning error that may occur when a robot attempts to execute a planned grasp. It is important that systems intended for real-world use be robust to these real-world imprecisions, so we discuss some methods developed to mitigate their effect. Finally, we discuss failure cases common across different approaches to grasp synthesis, and indicate the underlying shortcomings that they suggest. This leads naturally to my concluding suggestions for future research.

4 – Model

There are various approaches to the problem of choosing a single “best” grasp from a large set of candidates. This has been framed in the literature variously as a problem of classification, of ranking, and of regression. Perhaps the most intuitive method is binary classification on individual grasps. Not only is the problem easy to construct, it is also comparatively easy to create a training dataset for it. Labels are simple booleans, rather than, say, a real-valued quality metric that is difficult for a human to estimate. Boolean classification was used by one of the earliest papers in grasp learning ([9]), and still sees wide use in current work.

More recently, some researchers have proposed alternative ways to frame the problem. As motivation, they give specific rationales for which aspects of grasping performance are most important. In their 2016 paper, Gualtieri et. al. ([10]) articulate a clear objective: to maximize recall at high precision. Recall here is defined as the ratio of true positives found to all positives that exist — that is, the fraction of good grasps detected by the system. Precision is the ratio of true positives

<table>
<thead>
<tr>
<th>data ratio</th>
<th>Bottles</th>
<th>Small</th>
<th>Medium</th>
<th>Large</th>
</tr>
</thead>
<tbody>
<tr>
<td>Forest</td>
<td>0.83</td>
<td>0.43</td>
<td>0.45</td>
<td>0.31</td>
</tr>
<tr>
<td>CNN</td>
<td>0.83</td>
<td>0.39</td>
<td>0.51</td>
<td>0.41</td>
</tr>
<tr>
<td>OURS</td>
<td>0.85</td>
<td>0.59</td>
<td>0.70</td>
<td>0.84</td>
</tr>
</tbody>
</table>

Fig. 4: relative performance of the ranking CNN (“OURS”), a conventional classifier CNN, and a baseline implementation using random decision forests. The test was conducted for three size-based object classes, and one category class (“bottles”). The metric reported here is the ratio (true positives + true negatives) : (all classifications). The ranking CNN outperforms in all categories, by a wide margin in the case of medium and large objects. (Source: [11])
found to all positives found—the fraction of grasps attempted that actually succeed. Having high precision is the top priority, as failed grasps can be quite costly. Without compromising on that, we also want the highest possible recall, to come up with as many potential grasps as possible (in case, e.g., some grasps are impossible to execute due to obstacles in the workspace).

Kappler et. al. ([11]) uses that same argument as the rationale for a new ranking-based grasp synthesis system. Specifically, their system takes hypothesis grasps as input, and outputs a “top-1” ranking between a single “best” grasp, and all of the others (which are not ranked among themselves). This, they argue, makes intuitive sense, because grasp synthesis is ultimately concerned with picking a single grasp: the top one. They substantiate this claim by comparing the results of classifier and ranking Convolutional Neural Networks (CNNs) trained on the same dataset (see fig 4). Kappler et. al. attribute the performance increase to improvements in handling difficult and ambiguous grasps. The ranking system discounts these outright in favor of higher-ranked grasps which are unambiguously good. A classifier, on the other hand, determines the quality of grasps in isolation, without considering others.

4.b: Machine Learning Algorithms

Choice of learning algorithm has a profound impact on performance, and informs the design of other aspects of the system. Early work by Saxena et. al. ([9]) used logistic regression to perform classification. Subsequent work also made use of support vector machines [8, 12]. However, in the past three years deep-learning based methods have come to prominence. This parallels the broader rise of deep learning, which has been shown to achieve state-of-the-art performance in applications from visual recognition to natural language processing. The advantages of deep learning include its ability to learn useful feature representations directly from the data, and to learn complex nonlinearities in large scale datasets.
Kappler et. al. ([13]) demonstrate the value of deep learning methods, especially when large datasets are available. They perform a comparative test of a CNN and logistic classifier, trained on a dataset of approximately 300k grasps. As seen in fig. 5, the CNN consistently outperforms the logistic classifier, often by a wide margin. Another comparative test is conducted by Lenz et. al. ([14]), who compare the performance of a deep neural network to that of the linear SVM (from [8]). They, too, demonstrate a major improvement: the SVM-based method achieves an 84.7% success rate, the neural net 93.7%. It is notable that the training dataset was much smaller than that of Kappler et. al., and yet the performance improvement was still significant. Given the benefits of deep learning methods, it is no surprise that they have been used frequently in recent work ([15, 16, 17, 18]).

5 – Grasp Parameterization & Representation

To train a model for grasp synthesis, one must define a way to represent grasps in terms of the input sensor data. This must be a mapping between the grasp parameter space (e.g., a gripper’s 6D pose
\footnote{Three dimensions describing position (x, y, z) and three orientation (roll, pitch, yaw).}) and the input data (e.g., an image and/or point cloud), allowing different grasps to be distinguished. It must also modify/postprocess the input data to facilitate training; this varies

---

**Fig. 5:** Performance of a CNN and logistic classifier trained on the same grasp dataset. Each ROC is plotted for a false positive rate less than 20%, as we are only concerned with performance at high precision. Plots a, b, and c show results for small, medium, and large objects; d shows results for all objects. Note that “e-cnn/lreg” and “p-cnn/lreg” indicate the grasp quality metric used for training (ε and physics-based, respectively). “r” indicates the ratio of positive to negative examples in the datasets. (Source: [13])
depending on the machine learning technique used, but broadly requires extracting useful features in the data and reducing the data’s complexity to a tractable size.

5.a: Specifying a grasp

Most representations specify a grasp by including only data from that grasp’s immediate context. For example, Jiang et. al. ([8]) define a grasp for a parallel jaw gripper as a 2D rectangle in the input image. The rectangle’s position and rotation determined the grasp’s position and orientation, and the rectangle’s width determines the opening distance of the gripper’s jaws (see fig. 7). To recover the 3D gripper pose from that information, the center point is projected into the scene and the gripper is aligned along the normal vector at that point. Jiang et. al. propose this as an improvement over the representation used in [9], which merely labeled the center point of a grasp. Unlike that representation, the rectangle specifies a unique grasp, takes into account the dimensions

* Additionally, the rectangle’s length (blue) can indicate either the physical width of the jaws, or (for thin jaws) a range of acceptable grasping positions.
The rectangle representation has seen repeated use in the literature [14, 15, 17].

The rectangle representation is attractive because it is easy to implement, and provides a clear representation of a parallel-jaw grasp. However, it also restricts the space of possible grasps hypotheses from 7DOF\(^5\) (the gripper’s position, orientation, and opening width) to 4DOF (the grasp’s center point in the image, rotation about the object normal, and opening width). This undoubtedly excludes many valid grasps. Gualtieri et. al. ([10]) propose a representation which can represent grasps in full 3D (see fig. 6). Like the rectangle representation, their approach only considers input data between the jaws of the gripper; in this case that is a rectangular prism with points from the object’s point cloud. The prism of points is then discretized to a voxel grid.

### 5.b: Designing with Learning in Mind

The design of a grasp representation must take into account the limitations of the machine learning model and its training process. For example, the raw point cloud described above could be used as input to a neural network, but Gualtieri et. al. instead process it further to derive a 2D representation. The point cloud is viewed from three orthogonal perspectives. For each perspective a 2D image is made, with three channels of information: occupancy (the presence of

---

5 Degree of freedom
points), occlusion (whether each voxel was observable to the 3D sensor), and surface normals. This conversion is done primarily to reduce the number of input parameters to the network, for faster training and operation.

Additionally, the space of possible grasps parameterized by the representation must not be too large. This is of special concern for dexterous manipulators, where numerous degrees of freedom result in a configuration space infeasible for exhaustive sampling of candidate grasps. Varley et al. ([16]), whose experiments utilize a 7-DoF BarrettHand ([20]), note that the vast majority of hand configurations are not useful for grasping. They analyze a large corpus of grasps generated in simulation and find the eight most common (above a certain quality threshold). These “canonical grasps” are then used as labels for the training dataset. Instead of sampling a large space of possible grasps, the system instead samples image patches (representing the placement of one of the hand’s fingers; see fig. 8) and estimates the likelihood that a canonical grasp would be successful there.

A grasp representation must also present useful features for learning. For example, the 3D representation from [18] includes the visibility and occlusion of points in the grasp region (as viewed by the RGBD sensor). However, different machine learning algorithms require different levels of feature engineering. For example, a major benefit of deep learning techniques is their ability to learn feature representations. These improvements are noted by Lenz et al. [14], who use sparse autoencoders to pre-learn features for a DNN classifier. They draw direct contrasts with earlier work [8, 9], which used SVMs and logistic regressors requiring hand-engineered features.
6 – Dataset Generation

A fundamental problem in supervised learning is how to create the large datasets needed for training. This problem is especially acute in grasping, as the collection of example grasps on a real robot is expensive and laborious. Many different approaches have been taken in the literature, varying in how they choose and evaluate candidate grasps. An overview of some notable approaches is presented in Table 1. An important distinction between approaches is whether or not they require human supervision, as this is the primary bottleneck for dataset size. Papers which use manual labeling typically have datasets on the order of 500 to 5000 grasps, while papers which automate dataset generation end-to-end often demonstrate datasets orders of magnitude larger. I will discuss some representative approaches from the literature, and examine their relative advantages and limitations.

6.a: Supervised and Unsupervised Methods

In a recent paper by Goins et. al. ([19]), grasps are planned manually by humans and then manually reconstructed in a robot trial. Planning takes place in a simulation environment, where volunteers manually specify the position, orientation, and finger spread of the hand (a three-fingered BarrettHand [20]) relative to the target object. All grasps are then evaluated on an actual arm and BarretHand. This is a painstaking process, as object’s must be positioned precisely to match the values hard-coded in the system, or else grasps will be executed incorrectly. The resultant limitations of this method are clear: given a team of 22 volunteers, a total of 522 grasps were planned. Additionally, the authors note that volunteers had a preference for using power grasps rather than precision grasps, and also planning grasps based on semantic cues such as handles. While these grasps often perform well in practice, this still biases the dataset and may leave out grasps which are unintuitive to humans but work well†.

* It should be noted that the data gained from the evaluation was much richer than mere success/failure. A full set of analytical quality metrics were calculated (as is discussed in a subsequent section) as ground truth for the estimations made in simulation. However the observations on the limitations of robot trials and human annotation still apply.

† Consider, for example, a parallel jaw gripper which grasps a mug more reliably by its rim than its handle.
In contrast, Pinto & Gupta ([17]) demonstrate a fully automated approach to generating and evaluating grasps on a robot. This system is implemented on a Baxter robot, with a cluttered workspace of target objects on which to run grasping trials. Given an overhead view of this workspace, the system finds regions of interest (to limit the number of grasps executed where there are no objects), and uniformly samples grasps (using the rectangle representation from [8]). Grasps are then executed, with success inferred from the gripper’s force sensors. Grasps from this initial trial-and-error phase are then used to train a grasp synthesis model, which is in turn used to plan further test grasps; the model plans grasps with a much higher success-to-failure ratio, allowing more positive examples to be generated in less time. Over the course of 700 hours (often running more than 8 hours at a time), this system generated more than 50k examples.

While the benefits of Pinto & Gupta’s approach are undeniable, it also has its limitations. For example, it is doubtful how well trial-and-error grasp sampling would work for more complex grasp representations, such as the 3D representation from [10]. If the number of trials needed to find positive examples increases exponentially for these representations, then executing trials on-robot becomes infeasible due to constraints on available robot time.

6.b.: Advantages of Simulation

A popular approach to generating large datasets when labor and hardware are in short supply is through the use of a grasping simulator. Simulators such as GraspIt! [21] and OpenRAVE [22] support analytic methods for synthesizing grasps, including the estimation of many common quality metrics (such as ε- and volume quality). They additionally support limited dynamics simulation. Kappler et. al. ([13]) provide an example of the scale of datasets that can be created in simulation. They plan over 300k grasps, on over 700 CAD models (sourced from the 3DNet dataset as well as online). These grasps are generated in full 3D (i.e. not constrained to a single viewpoint). First, surface points are sampled by casting rays inwards from the sides of the object’s bounding box; grasps are then oriented along the surface normal at those points, and attempted for a number
of different rotations about the normal. An alternative method for grasp synthesis is to optimize over a quality metric. GraspIt! does this, using simulated annealing to optimize for a variety of common metrics, and in several different search spaces (e.g. approach vectors parallel to surface normals).

6.c: Limitations of Simulation

Although simulation avoids some of the problems of both manual labeling and robot trials, it has problems all its own. First and foremost, because it is based upon analytic methods, it suffers from their shortcomings: the predicted quality of simulated grasps is not always correct. Goins et. al. ([19]) assess the usefulness of many metrics by recreating simulated grasps on a real robot (as described in 6.a) and comparing the groundtruth quality metrics to those estimated by the simulator. They find that with the exception of the $\varepsilon$-metric, most metrics perform rather poorly. Kappler et. al. ([13]) provide further analysis of the $\varepsilon$-metric in their 300k-grasp dataset, finding that while thresholding on a very small value does filter out many bad grasps, the $\varepsilon$-metric does not provide a clear separation between good and bad grasps. This problem, they assert, becomes even greater when comparing two different objects, even when they are of the same category.

As an alternative Kappler et. al. propose a new “physics metric” for determining grasp stability. To calculate this metric, the object is grasped and then subjected to a series of small perturbations in pose. After each perturbation, the object is checked for collisions with the hand. If no non-colliding perturbations are found, then the object is determined to be stable. By comparing this metric to the $\varepsilon$-metric, and evaluating both through crowd-sourcing, they find the physics metric to have improved performance.

Another issue with simulation is the difference between real and synthetic sensor data. For example, synthetic images are not yet “photoreal”, and statistical differences between real and rendered data may cause models trained synthetically to lose performance when applied to non-synthetic data. In addition to the obvious example of rendering color images of 3D scenes, depth
sensors (such as the Kinect) introduce noise into their measurements which must be modeled when simulating them. Kappler et. al. introduce random perturbations into point clouds to simulate this. In general, though, problems with synthetic images are more severe with color than with depth, as complex lighting and surface reflectance properties must be simulated to properly render texture. This challenge extends to finding 3D object scans with high-fidelity textures; at the time of writing options are few, with the BigBIRD dataset ([23]) providing fewer than two hundred models.

7 – Directions for Future Research

Based on the research reviewed in the preceding sections, I will now identify some of the current unsolved problems in grasp synthesis, and propose concrete directions for future research.

7.1: Dataset Generation Techniques

With the rise of deep learning for grasp synthesis, it is more important than ever to have large datasets of grasp examples. I believe that grasp simulation and self-supervised grasp learning techniques are the most promising approaches to generating these datasets. Based on the information reviewed in this paper, we propose two immediate subjects for future research. First, we recommend that the self-supervised learning techniques introduced by Pinto and Gupta ([17]) be extended to more complex grasp representations. The rectangle representation they use is straightforward, but limited; we believe that future advances in grasping will utilize representations which permit grasps to be sampled more freely. Second, we recommend further development of grasp quality metrics which utilize dynamics simulation. The work of Kappler et. al. ([13]) demonstrates the superiority of their dynamics-based “physics metric” over static metrics (in this case the $\varepsilon$-metric). Developing and improving these metrics will make simulation more useful.

7.b: Accounting for Object-level Properties

I also believe that the current paradigm of grasp synthesis systems has fundamental limitations. Namely, because these approaches learn only local “graspability” features (based on
image texture and geometry within the input region), they fail when the grasped region has unexpected material properties, or when the grasp is in a suboptimal position on the object. For example, they might choose to grasp a long baton at its end, resulting in torques which wrench it free; or they might attempt to grasp shoes by the laces, only to have them slip away. To avoid these failure cases, new approaches must be developed which take these factors into account.

An elementary example of such a system could define regions of interest for grasping on an object, based on insights into its structure or material properties, and then determine exact grasps from within these regions using a local feature-based grasp synthesis system. A similar approach has been demonstrated recently by Song et. al. ([24]), who combine a local-feature based grasp synthesis system with a “global” system which classifies an object, estimates its pose, and estimates the location of grasp affordance regions.
Part 2: Learning “Sensible” Grasps

With a Novel Application of Object Pose Detection and Synthetic Training Data

1 – Introduction

As I have discussed, current systems for learning-based grasp synthesis fail to see the big picture. They judge grasp quality using using only texture and structure information within a small region of interest (for example, the rectangular region between the jaws of a parallel jaw gripper), missing potentially critical object-level information. Seeking to address this, I propose an approach to grasp synthesis which is quite different from the “grasp rectangle classifier” paradigm discussed thusfar. My system combines recent advances in 3D object pose detection with a set of tools to facilitate its application to grasping, allowing the system to identify the 3D location of sensible grasping points for an object. Moreover, it does this from a single 2D image, without depth information, making it robust to sunlit conditions (e.g., outdoors) where RGBD cameras have degraded performance.

My object pose detection system, based on work by Pavlakos et. al. [26], builds upon recent advances in applying convolutional neural networks (CNNs) to the 3D object pose detection problem.
Given a training dataset of reference images annotated with an object’s 3D pose, I train a CNN-based system to estimate an object’s 3D pose in new images. To use this for grasping, I assume knowledge of predefined grasping points on the target object; once I know the object’s world pose, I can find the world pose of the grasping points.

However, this raises the problem of defining where the good grasping points are for a given object. I propose to identify good grasping points by conducting robot grasping trials with human assistance, and demonstrate a proof-of-concept implementation. I believe this approach to be a reasonable “first step”, making necessary compromises between ease of implementation, ease of use, and performance. Because this step is labor intensive, I attempt to offset that by creating new tools for generating synthetic datasets for the object pose detector. I hope that these can be used to experiment with training object pose detectors more broadly.

3D pose detection systems have only recently been developed which are accurate and robust enough for grasping. In reviewing the literature on both grasping and pose detection, I found only passing mention of this application. Thus I also discuss the considerations and problems I faced in designing and implementing my system, and reflect both on practical improvements to my current design and larger conceptual questions about the untapped potential and possible limitations of using object pose detection for grasping.

2 – Related Work

In this section I will first focus on pose detection, defining the problem and reviewing past and current trends in research on the subject. I will then compare aspects of my pose detection-based grasping system to existing work in the grasping literature.

2.a: Object Pose Detection – Definition and Key Challenges
The goal of object pose detection is to extract information about an object’s position, orientation, and (sometimes) structure from a single RGB image. Some systems estimate only 2D information, while others attempt to infer 3D information as well. There are two distinct areas of research, one focusing on human pose detection (and consequently receiving somewhat greater attention) and the other on non-human objects. They bear distinction because humans are much more articulated than objects, and thus the two have different requirements: objects can usually be described with a single 6-dimensional pose, while humans require the estimation of many different joint parameters. I focus on object pose estimation, but the two fields of research are highly interrelated and I will draw upon the literature for both.

Typically, the problem is framed as one of localizing in the image representative “keypoints” of the object in the image, which are then used to determine the object’s 3D pose (fig. 2). For pose detection on objects, these keypoints usually correspond to known points on a 3D model of the object (for humans, joints). Desirable qualities for a pose detection system are robustness to clutter (which can occlude keypoints, modify the object’s apparent structure, etc.), variations in object texture (due to lighting or innate variability for that kind of object), and variations in object geometry. A key challenge in designing learning-based approaches is the difficulty of creating training datasets with groundtruth 3D poses. For example, the Pascal3D+ dataset [27] was generated by manually identifying image-model keypoint correspondences.

**fig. 2** An example of keypoint detection for estimating human pose. *(Source: [33])*
Earlier work in pose detection centered around template matching. These systems were somewhat limited to single instances of a given object, as well as very limited changes in camera viewpoint. Attempts to address these shortcomings included using multiple different templates for different object textures and viewpoints (for example, spaced around a viewing sphere [24, 28]).

Another representative approach from the pre-deep learning literature is Zhu et al. [29] (see fig. 3). It attempts to avoid pitfalls of appearance-based approaches (including those based on template-matching) by using a shape representation based on the object’s silhouette. Thus it is more robust to variations in appearance due to lighting, object texture variability, and related factors. Additionally, the system demonstrates robust segmentation in clutter without depth information, allowing the system to operate in realistic outdoor settings.

Zhu et al.’s system first estimates a bounding box for the object, using a Deformable Parts Model (DPM)-based classifier which is given image contours as input (fig. 3b). This step also determines a coarse representation of the object’s pose, which is later refined. The next steps attempt to extract the object’s silhouette: first, the cropped image is segmented into “superpixels”, some combination of which have a high probability of corresponding to the silhouette (fig. 3c). A reference silhouette is generated using a known 3D model of the target object and the pose estimate given by the
DPM. This reference silhouette is then used to find the optimal superpixel combination; in this step, reference and candidate silhouettes are also converted into a shape descriptor called a “chordiogram” to aid the matching process. Finally, the initial pose estimate is refined such that the reference silhouette better matches the one extracted from the image (fig. 3d). Zhu et al. also introduce a new 3D pose detection dataset of outdoor and cluttered scenes, highlighting the strengths of their RGB-only system compared to those dependent on depth cameras. Lastly, they also demonstrate the system’s potential application to grasping.

2A – Rise of CNN-based Approaches

A major shift in pose detection research came with the introduction of deep convolutional neural networks (CNNs), which since 2012 have seen wide application to problems in computer vision. Toshev and Szegedy [30] were, in 2014, the first to apply CNNs to human pose detection. Notably, their system significantly outperformed then-state-of-the-art DPM-based systems, despite using a “generic” network architecture originally designed for image classification ([31]). Tompson et. al [32] applied purely convolutional neural networks to human pose detection, again improving the state of the art by a wide margin. Importantly, their approach utilizes a CNN jointly with a graphical model which learns the typical structural configuration of human joints. This was done because the CNN, designed to perform keypoint localization, was found to be especially prone to false positives where the relative positions of joints were anatomically nonsensical. The graphical model, trained to estimate the likelihood of a joint being in a given location relative to the other joints, was demonstrated to be quite effective in pruning false positives.

Newell [33] demonstrated a purpose-designed CNN architecture for 2D human pose detection. His “stacked hourglass” is designed to incorporate multi-scale inference without the need for a distinct graphical model. The network is comprised of multiple “hourglass” modules, with the output of one fed into the next. The hourglass module (see fig. 5) design uses symmetric downsampling and
upsampling to force the network to consider multiple feature scales. Beginning at full resolution, the image is successively downsampled until, by the middle layers, it is a mere 4x4 pixels; the image is then upsampled back to full resolution. To prevent the loss of information during downsampling, earlier layers are connected to the later ones via skip layers. The effect of this design is to essentially integrate reasoning on multiple scales of features within the image: the high resolution first (and last) layers may expose information about the appearance of individual body parts, while the middle layers prioritize more global structure (e.g., posture).

The network is comprised of multiple hourglass modules stacked end-to-end (fig. 4), with intermediate supervision applied at the output to each module. This allows for the refinement of joint location estimates, and helps to guarantee a strong gradient for training throughout the network, avoiding the “vanishing gradient” problem. Newell demonstrates a clear performance improvement in using the stacked hourglass architecture, and in using intermediate supervision.

---

1 The “vanishing gradient problem” describes the tendency of each layer in a neural network to train much slower than its successor. For more information, see Hochreiter et al. [37].
Pavlakos et. al. [26] adapts the stacked hourglass architecture for use with general objects, and adds a 3D pose estimation step based on a constrained optimization. Notably, they also expand the system to class-level pose detection, using a deformable shape model to accommodate variations in structure in a class of objects. The deformable shape model for a given object class is derived from a set of CAD models representative of the class’ variability. Given keypoint annotations for the models, principle component analysis is performed to extract a set of representative deformations on keypoint location. Objects of a given class can be approximated as a weighted sum of those representative deformations.

Deep learning methods demand the creation of larger training datasets, which (as mentioned previously) is a longstanding challenge for 3D object pose detection. Wu et. al. [34] introduce a novel method to circumvent the need for 2D-to-3D image-to-model correspondences in training. Their network, trained for a specific category of objects (with structural variation represented using the same method as [26]), regresses directly from keypoints to the object pose and structural parameters. This neural network has two distinct components which are trained separately. The first performs keypoint identification in 2D, trained with images and keypoint labels (as in [26]). The second, called the “3D interpreter”, regresses from these keypoints to the object’s pose and structure. Instead of using 3D groundtruth labels for training images, the “3D interpreter” stage is trained with synthetic data: keypoint location heatmaps (the training input) generated from 3D models placed in known poses (the training label). This clever system has all the advantages of synthetic data (data is practically free) with none of the downsides (in this case, limited photorealism).

---

2 This is a simplification– the network additionally has a keypoint refinement stage between keypoint estimation and pose regression.
3 – Materials & Methods

My system has three distinct components: the object pose and grasp point detector, the robot-based dataset generation system, and the synthetic data generation system. I will address these separately.

1 - Object pose and grasp point detector:

I use the CNN-based object pose detection system from Pavlakos et al. [26]. As I discussed, it uses a stacked hourglass module design for 2D keypoint estimation (with intermediate supervision at each module), and then estimates 3D pose and structure simultaneously by minimizing the reprojection error of the keypoints of a reference model. I chose this system as it has (at the time of experimentation) state of the art performance for 3D object pose detection. It has also been demonstrated to achieve this performance using relatively small training datasets, which is a benefit in general but was especially desirable for my small-scale experiment.

2 - Dataset generation on a robot, and grasp point identification.

My goal with this system is to generate a dataset which can be used to identify where the successful grasping points are on an object. My approach is to manually guide the robot to execute a series of trial grasps, recording data describing the circumstances and outcome of those grasps. For my experiment, I used the arm of a Baxter Research Robot, outfitted with the standard parallel jaw grippers\(^3\). As a target object I chose a plastic gas can (see fig 6), as it demonstrates a use-case for globally-informed grasp synthesis: grasps made along the handle may vary in stability depending on

---

\(^3\) Specifically, the Electric Parallel Grippers with 112m long fingers and half-round fingertips.
their distance from the center of mass, even though local geometry and texture does not change significantly.

The minimum information needed is the pose of the gripper relative to the target when grasping, and a positive/negative label for the grasp’s success. I also stored color images of the object, for use as training examples for the pose detection system. Object and gripper pose labels were recovered from RGBD images taken by an Asus Xtion sensor. Given a reference model of the object (fig 6), the ground truth object pose is found by matching the model’s geometry to the Xtion’s point cloud, using the Iterative Closest Point algorithm. To find the gripper pose, I use an array of AprilTag fiduciary markers ([35]) mounted to the robot’s final forearm segment. These markers give a more accurate position estimate than what can be derived from the Baxter’s internal joint encoders.

To generate my dataset, I placed the gas can on a table in front of the robot, and manually executed and labeled a series of grasps. I attempted grasps with the object in several different orientations, to ensure a representative sample of the object’s appearance in the pose detection dataset. For each orientation, I attempted grasps at points along the gas can’s handle, spaced approximately four centimeters apart. At each point, I attempted to grasp with the gripper at three orientations: parallel to the table, forty-five degrees above the table, and ninety degrees above the table (a standard “pick-and-place” grasp). Grasping was attempted by closing the gripper’s jaws and moving the hand vertically upward: if after five seconds the object showed no signs of dropping or shifting significantly, the grasp was considered a success. This process was facilitated by a set of tools I created for recording, processing, and visualizing grasp attempt data.

These are available here: https://github.com/JSR694/grasp_attempt_dataset_tools
3 – Synthetic training data for pose detection.\(^5\)

My goal was to create a system which could automatically generate large training datasets of labeled images for the object pose detector. I wanted these images to reflect real-world challenges such as clutter and variation in background texture and lighting.

For my trial dataset, I used a gascan model generated from 3D scans of an actual gascan using a Kinect RGBD sensor and the Skanect software\(^6\). I also recreated the object’s simple texture manually in Blender\(^7\), and specified a set of discriminative keypoints. In practice, my system accepts any Gazebo-compatible 3D model. Images are rigged and rendered in the Gazebo robotics simulator\(^8\), which uses the OGRE rendering engine\(^9\). This offers easy integration with the rest of my system (via Gazebo’s ROS interface), and decent rendering quality.

To create a training example, the target object is placed in a random pose (the simulated camera is static) and additional objects are spawned randomly near the object as clutter (if that is specified). These objects are from the BIGBird dataset [23], and have realistic high-resolution scanned textures and meshes. When the scene is rigged, a simulated camera captures an image, which is stored in an HDF5 dataset along with the object’s pose. This process can be repeated to create more examples as needed. However, due to the limitations of the current Gazebo API, changes to lighting and background must be made by editing the world definition file and re-running the system.

\(^{5}\) Code available here: [https://github.com/JSR694/model_pose_dataset_generation](https://github.com/JSR694/model_pose_dataset_generation)
\(^{6}\) [http://skanect.occipital.com/](http://skanect.occipital.com/)
\(^{7}\) [https://www.blender.org/](https://www.blender.org/)
\(^{8}\) [http://gazebosim.org/](http://gazebosim.org/)
\(^{9}\) [http://www.ogre3d.org/](http://www.ogre3d.org/)
4 – Results and Discussion

I will now discuss the results of tests carried out of the tools and protocols for dataset generation discussed in the preceding section. I will then reflect on the problems encountered during these tests, and how they could be solved for future experiments.

fig. 7: Examples of images created by my synthetic dataset generation system.

1 – Synthetic Data Generation:

My current method of synthetic data generation was able to generate 10 training examples in one minute when spawning objects with clutter. This puts it well within the rate of productivity needed to be useful for generating training datasets for pose detection. And given that the program can generate data for an arbitrary amount of time without user supervision, it can generate arbitrarily large training datasets.
2 – Results: Grasp attempt dataset

In thirty minutes of data collection, I were able to collect 40 grasp examples. Of those, 26 were labeled “good”, 4 “bad”, and 10 “invalid”. Grasp attempts were “invalid” if the target object was accidentally moved between capturing the object pose and gripper pose data. Note that sensor data for “invalid” grasps was still used in the pose detection training dataset.

While my particular experimental setup could have been more efficient, the rate of data generation is more than sufficient for generation pose estimation datasets: [Pavlakos 6Dof] demonstrated cutting edge performance with a dataset of just 150 examples. However, it is still unsustainable if I need to collect examples for many different object instances– for example, when training a class-level pose detector.

Despite the small sample size and overwhelming success of the grasps in my dataset, I believe the method produced valid insights into which grasping points were good and bad. For example, three of the four bad grasps were made with a vertical (“pick and place”) gripper (see fig 8). At least one was made towards the read of the handle, resulting in an imbalanced grasp which wrenched the gascan loose of the gripper. Meanwhile, only one bad grasp was conducted with the gripper parallel to the table (this is because the gascan handle is resting on the lower gripper, and thus cannot slip away). I believe that given a larger dataset, grasp quality would be represented well.

![fig. 8: Examples of two failed grasps (with point cloud visualized at right).](image)
3 – Challenges and Future Improvements

The process of testing my grasp attempt dataset generation system revealed a number of practical and technical problems which reduced the amount and quality of the data I collected. I will now discuss them, and propose solutions for future experimentation.

**Improving accuracy of groundtruth pose:** the iterative closest point algorithm frequently produced slightly misaligned groundtruth object poses. This is a significant problem, as pose detection accuracy is paramount for my detected grasps to be successful. This error is a consequence of the point cloud map of the object being too noisy and/or sparse to match the reference mesh well. For example, significant pose estimation error correlated strongly with the gascan being oriented toward the sensor, as that caused significant structural features such as the gascan’s spout to be missed (see fig 9 for a comparison).

**Improving the speed and reliability of grasp attempts:** my current system relies on the user to enact all movements of the robot’s arm. This takes a significant amount of time, and could be partially automated. For instance, the process of moving the arm out of the camera’s view (for the object pose groundtruth data) and of executing grasp. Additionally, the current mounting of the camera array between the robot’s arms results in frequent collision and misalignment of the cameras, which takes time to fix and results in invalidated grasp attempts.

---

10 Note that I have few recommendations for improving the synthetic dataset toolchain at this time. My primary difficulty in developing and using it was the general unreliability of the libraries used. I may switch away from the Gazebo simulator to a different simulator/render in the future.
Limited functional workspace due to field of view restrictions:
The robot’s workspace is currently quite limited by the positioning of the cameras and by my visual fiduciary system. Objects to be grasped cannot be placed too close to the robot, or the robot’s first arm joint will collide with the camera rig; too far and the AprilTag marker leaves the camera frame.

Common solution - reconfigure sensor setup: All of the above could be remedied by adding additional cameras and depth sensors to perform groundtruth data collection, positioned outside of the robot’s work envelope. For instance, two depth sensors on either side of the robot’s workspace would ensure that the object was never severely occluded, allowing for improved ICP localization. To maintain the first-person perspective for training images, a webcam could be mounted in place of the current bulky RGBD sensors, reducing the risk of collision with the robot’s arms.

5 – Conclusion

In this paper, I proposed using object pose detection as a method for planning grasps based on global object properties. To further this, I developed a set of tools for determining grasping points on objects and for generating large training datasets to make object pose detectors easier to train for new objects. I demonstrated the functionality of both of these systems, and made practical recommendations for their improvement. Though currently limited, both show promise to deliver on their stated purposes.

My next course of research will be to put these into practice (once the aforementioned improvements have been made), using them to train an object pose detection system for grasping. My
The greatest uncertainty is whether the object pose detection system (from [27]) is accurate enough to outperform local-feature-based systems. In [27], centimeters of error were detected on average for the gascan instance (i.e. not class-level) object pose detector. It may be the case that my grasping system is just outside threshold of accuracy for recommending exact grasps; in which case, I may modify the system to use the pose detector to generate regions of interest, and then use a local feature-based grasping system to plan the final grasp.

In the future, I would also like to explore other applications of the object pose detector’s ability to recover 3D information from 2D images. One could envision, for example, gathering large datasets of human manipulation actions from unlabeled online videos. By tracking the type of grasp a person uses (as done in [36]) to grasp an object, and tracking where and how they manipulate that object, one could infer information about how objects are used and the physics of grasping.

Acknowledgements

I would like to sincerely thank Professor Daniilidis for giving me the opportunity and support to explore a subject of research that has since become a personal fascination; Georgios Pavlakos and Alex Zhu for their invaluable guidance in designing my experiment and technical support in carrying it out; and Professor Lindell and the Haverford Computer Science department for their trust, support, and patience as I have pursued this project off-campus.
References
(with annotations)

   - An example of a direct application of pick-and-place grasping.

   - The definitive survey of data-driven approaches to grasp synthesis. Provides a solid overview of analytical techniques as well.

   - A standard reference for analytical grasping.

   - Comprehensive reference on robotic manipulation. Section 7.1 gives great succinct overview of analytic grasping fundamentals.

   - In-depth review of analytical grasping concepts. Definitions of force-closure, form-closure, grasp wrench space, etc.

   - Whitepaper for the GraspIt! Grasping simulator. Contains useful definitions for all analytical techniques employed in simulation.

   - Provides good overview of some fundamental analytic grasping terms. Source for a useful illustration.

   - Introduced a widely-used grasp representation and training dataset.

   - Seminal paper on learning for grasp synthesis. Referenced heavily by subsequent work.
   • Proposes an interesting new grasp representation. Studies the impact of incorporating category- and object-level knowledge into model training. Also, studies the usefulness of pre-training a grasp synthesis CNN with RGBD data synthesized from CAD models.

   • Aims to reframe the grasping problem as one of ranking. Proposed mathematical methods for training to rank with only boolean training labels. Demonstrates improved performance in a CNN-based ranking system compared to a classifier CNN trained and tested on the same data.

   • Introduces a novel grasp representation allowing grasps to be sampled in 3D. Proposes a useful geometric heuristic for choosing candidate grasps for parallel-jaw grippers.

   • Demonstrates the advantages of deep learning for grasp synthesis. Proposes a novel physics-based quality metric for grasp simulation, and determines its usefulness compared to existing metrics. Generates a large dataset of example grasps in simulation, using this metric.

   • Applies deep learning to the grasp synthesis problem. Proposes a cascaded network architecture, and pre-trains network using features learned with sparse autoencoders. Conducts extensive tests, of both learned models and full grasp synthesis systems on robots.

   • Recent application of CNNs to grasp synthesis. Also proposes new method for picking the single “best” grasp when presented with many positively-classified options.

   • Applies deep learning to learn grasps for dexterous hands. Presents a novel representation of grasps for multi-fingered grasps. Uses the GraspIt! simulator to generate grasps on scanned 3D models, and then renders them in the Gazebo simulator.

   • Novel approach to automating the generation of training grasps on a robot. Uses a multistage learning approach to improve its model as it is training it.
   • Proposes a new 3D grasp representation for point cloud data. Experiments with pre-training a
     CNN using simulated point cloud data derived from CAD models. Also experiments with
     category-specific training.

   • Grasp metrics, machine learning, simulation

20. Townsend, William. "The BarrettHand grasper-programmably flexible part handling and
   • An example of a commonly used dexterous manipulator.

   • Whitepaper for Graspit!, one of the most popular grasping simulators. Used in multiple papers
     cited in this survey.

22. Diankov, Rosen, and James Kuffner. "Openrave: A planning architecture for autonomous robotics."
   • A robotics simulator with popular extensions for grasp planning. Used frequently in the
     literature.

   • Dataset of textured 3d object scans. Used to create datasets of simulated grasps, in Varley et. al.
     among others.

   • Combines a Saxena2008-like grasp synthesis system with object classification, pose estimation,
     and affordance location estimation.

   • A striking example of applications of reinforcement learning to robotic control. This is a
     completely different approach to the grasping problem. It is useful as a comparison to the
     supervised learning-based systems covered in this review.

   • Example of an CNN-based system for object pose detection. The first such system to
     incorporate a deformable shape model, improving performance on class-based pose detection.
   • A commonly used dataset for 3D object pose detection research. Also provides a good example of the difficulties involved in building object pose detection datasets.

   • A representative example of template-based object pose estimation.

   • Example of a pre-deep learning system for object pose detection. Articulates many of the core problems in object pose detection, and highlights its potential usefulness in grasping.

   • The first application of deep convolutional neural networks to pose detection.

   • Landmark paper on applying CNNs to large scale object recognition. Precipitated a shift in the vision community toward the use of CNNs and other deep neural network architectures.

   • The first paper applying a pure CNN to human pose detection (the architecture in [30] incorporated other varieties of neural network). Demonstrated the utility of training a graphical model alongside the CNN.

   • Describes a new CNN architecture for 2D human pose detection. This design is the basis for the architectures used in Pavlakos2016 and Pavlakos2017, and thus also that of our system.

   • A novel object pose detection system, trained with a combination of of real and synthetic data. Demonstrates an interesting strategy for overcoming the dataset bottleneck, and uses deep learning for both 2D and 3D structure and pose estimation.

   • Whitepaper for the visual fiduciary used in our experimental setup for generating a grasp attempt dataset.
   • Representative of recent research on human grasp type detection and action understanding.

   • Provides a thorough explanation of the vanishing gradient problem in neural networks. This phenomenon is one of the motivations for the stacked hourglass architecture used in [33].