



University of Stuttgart
Institute for Control Engineering
of Machine Tools and Manufacturing Units
(ISW)



Master's Thesis

SLOG: Single Label Object Grasping

submitted by

Kanishk Umesh Navale

from India

Degree program
Examined by
Supervised by
Submitted on

M. Sc. Computer Science
Prof. Andreas Wortmann
Mr. Tobias Bux
31 October 2022

Declaration of Originality

Master's Thesis of Kanishk Umesh Navale (M. Sc. Computer Science)

Address Allmandring 22A-10, 70569 Stuttgart
Student number 3437531
English title *SLOG: Single Label Object Grasping*
German title *Einklassige verallgemeinerte visuelle Einbettung von Objektpunkten für das Greifen von Robotern*

I now declare,

- that I wrote this work independently,
- that no sources other than those stated are used and that all statements taken from other works—directly or figuratively—are marked as such,
- that the work submitted was not the subject of any other examination procedure, either in its entirety or in substantial parts,
- that I have not published the work in whole or in part, and
- that my work does not violate any rights of third parties and that I exempt the University against any claims of third parties.

Stuttgart, 31 October 2022

Kurzfassung

In dieser Arbeit wird ein Rahmenwerk für künstliche Intelligenz vorgeschlagen, um eine verallgemeinerte visuelle Objektdarstellung für Roboter-Greifaufgaben zu entwickeln.

Ein einzelnes Label wird aus der verallgemeinerten visuellen Objektdarstellung extrahiert, um ein Objekt auf einer pixelbasierten Skala zu identifizieren, das dann als Greifpunkt für einen Roboter dient. Die verallgemeinerten visuellen Objektdarstellungen, die aus Dense Object Nets (DON) berechnet werden, werden zur Berechnung der 6D-Position von Objekten verwendet, um eine robuste Roboter-Greifpipeline zu erstellen. Um die Notwendigkeit einer manuellen Extraktion von Beschriftungen aus den von DON berechneten Repräsentationen zu eliminieren, wird KeypointNet implementiert. Die Pipeline für die semantische Objektkorrespondenz wurde auf der Grundlage der Generalisierungsfähigkeiten des KeypointNet entwickelt, um DON weiter zu trainieren. Gleichzeitig haben wir festgestellt, dass das KeypointNet für sich genommen bereits die Fähigkeit besitzt, verallgemeinerte visuelle Objektrepräsentationen ähnlich wie DON zu berechnen, was den Gedanken nahelegt, dass es auch andere Netzwerke gibt, die in der Lage sind, verallgemeinernde Merkmale zu erzeugen.

Im Kern demonstriert diese Arbeit die eingeschränkten Fähigkeiten von Netzwerken für die Erzeugung verallgemeinerten visuellen Objektdarstellungen, wenn sie nur auf dem synthetischen Datensatz trainiert werden.

Im Gegensatz zum isolierten KeypointNet konnten wir für das KeypointNet Robustheit gegenüber Objektverdeckungen im Blickpunkt erreichen, wenn es auf DON-Darstellungen in einer End-to-End-Methode trainiert wurde. Darüber hinaus wählt das KeypointNet, das mit den von DON berechneten Repräsentationen arbeitet, selbstständig einzelne Labels aus und berechnet geometrisch konsistente 6D-Posen für semantisch ähnliche Objekte. Abschließend wird die Robustheit des objektgeneralisierenden Frameworks bei der Generalisierung von realen Kappen und Manipulationen anhand eines realen Robotermanipulators demonstriert.

Stichwörter: Künstliche Intelligenz, verallgemeinerte visuelle Objektrepräsentationen, DON, KeypointNet, geometrisch konsistente Keypoints, semantische Objektkorrespondenz-Pipeline, DON-Informiertes KeypointNet, Ende-zu-Ende-Training von neuronalen Netzen, synthetischer Datensatz, robuste 6D-Objektposen, Eizelnlabel-Objektgreifen, Demonstration von Roboteraufgaben

Abstract

The thesis proposes an artificial intelligence framework to develop generalized visual object representation for robot grasping tasks.

A single label is extracted from the generalized visual object representation to identify an object on a pixel-wise scale further acting as a robot grasping point. The generalized visual object representations computed from DON are applied to compute objects' 6D pose to create a robust robot grasping pipeline. To eliminate the need for manual label extraction from the representations computed by DON, KeypointNet is implemented. The semantic object correspondence pipeline is engineered based on generalizing capabilities of the KeypointNet to further train DON. At the same time, we identified that the KeypointNet on its own already demonstrates the capabilities of computing generalized visual object representation similar to DON, ushering the idea that there are other networks capable of the generalizing features.

At its core, this thesis demonstrates the reduced capabilities of networks to produce generalized visual object representation when trained on the synthetic dataset only.

In contrast to KeypointNet training in isolation, we could achieve robustness for the KeypointNet against the object occlusions in viewpoint when trained on DON representations in an end-to-end fashion. Furthermore, the KeypointNet informed with DON computed representations, autonomously picks single labels and computes geometrically consistent 6D poses across semantically similar objects. Finally, the object generalizing framework is demonstrated for its robustness in generalizing real-world caps and manipulation by employing a real-world robot manipulator.

Keywords: artificial intelligence, generalized visual object representations, DON, KeypointNet, geometrically consistent keypoints, semantic object correspondence pipeline, DON Informed KeypointNet, end-to-end neural network training, synthetic dataset, robust object 6D poses, single label object grasping, robot task demonstration

Acknowledgement

I want to thank the entire sereact team for their great support and guidance throughout completion of this thesis. Furthermore, I want to thank Olga Klimashevska (OK) and Nirbhay Jain (Bablu) for their beneficial input.

Contents

Kurzfassung	ii
Abstract	iii
Acknowledgment	iv
1 Introduction	1
1.1 Motivation	2
2 State of the Art	5
2.1 Overview	5
2.2 Breakdown Analysis & Comparison of Influential State of the Art (SOTA)	7
3 Objective	11
4 Methodology	12
4.1 Dense Object Nets	12
4.1.1 Implementation Strategy	13
4.1.2 ResNet Architectures	13
4.1.3 Loss Functions	13
4.1.4 $PCK@k$ as an Evaluation Metric	16
4.1.5 Suzanne, a synthetic dataset	16
4.1.6 Manual Method for Robot Grasp Generation	17
4.2 KeypointNet	18
4.2.1 Loss functions	18
4.2.2 Total Weighted-Sum Loss	22
4.2.3 Network Modification	23
4.3 End-to-End Network Training	23
4.3.1 KeypointNet Informed Dense Object Nets	23
4.3.2 Dense Object Nets Informed KeypointNet	24
4.3.3 Mining Dense Object Nets Representations in KeypointNet	24
4.4 Generalizing “Caps”	25
4.4.1 Semantic Correspondences Mapping Pipeline	26
4.4.2 Tweaking the Training Pipeline	26
4.4.3 Evaluating Single Class Generalized Labels in the Wild	27
4.4.4 Robot Grasping Pipeline	27

5 Results	29
5.1 Dense Object Nets	29
5.1.1 Manual Method for Robot Grasp Generation	30
5.2 KeypointNet	31
5.3 End-to-End Training	32
5.3.1 KeypointNet Informed Dense Object Nets	32
5.3.2 Dense Object Nets Informed KeypointNet	32
5.3.3 Mining Dense Object Nets Representations in KeypointNet	33
5.4 Generalizing “Caps”	35
5.4.1 Semantic Correspondence Mapping Pipeline	35
5.4.2 Evaluating Single Class Generalized Labels in the Wild	36
5.4.3 <i>“Not losing hope on the real world”</i>	37
5.4.4 Robot Grasping Pipeline	39
6 Conclusion	41
7 Future Scope	43
Bibliography	44
List of Acronyms	48
List of Figures	49
List of Tables	50
Listings	51
List of Symbols	52

1 Introduction

Creating a general-purpose robot that can carry out practical activities, like Chappie or C-3PO, is one of the objectives of robotics in general and robotic manipulation in particular. Even if advancements toward this objective have been made recently in adjacent domains, it is still a work in progress. E.g., AlphaGo [1], a gameplaying artificial intelligence system trained entirely on self-play, defeated Lee Sedong, the world's best human Go player at the time. Subsequently, Silver *et al.* [2], developed artificial intelligence algorithms mastering the game of chess, Go, World of Warcraft and Shogi, surpassing human playing expertise. Most of these algorithms learn by drawing understanding directly from visual data such as gameplay recordings or online video streams emphasizing that the visual data is essential.

Meanwhile, the launch of AlexNet [3] in 2012 transformed the realm of computer vision. Other visual tasks, such as semantic segmentation [4], object identification and recognition [5], and human posture estimation [6], witnessed significant gains in the years that followed. Robotics has also made significant breakthroughs, ranging from self-driving cars to humanoid robots capable of performing remarkably active jobs recently developed using camera and other vision sensors.

Despite these advancements, the most frequently used robotic manipulation systems haven't evolved much in the previous 30 years. Typical auto-factory robots continue to do repetitive operations such as welding and painting, with the robot following a pre-programmed course with no feedback from the surroundings. If we want to increase the utility of our robots, we must move away from highly controlled settings and robots that do repetitive actions with little feedback or adaptability capabilities. Liberating ourselves from these restraints would allow us to enter new markets, as has been witnessed by the proliferation of firms competing in the logistics domain.

In this thesis, we want to take things further by providing a generalized approach for vision-based robot tasks in real-world contexts. We employ a combination of classical robotics, computer vision and current deep-learning methods. While there are lessons to be drawn from deep learning, artificial intelligence and computer vision breakthroughs, we highlight the particular problems of machine vision based problems in robot manipulation. We offer an innovative generalized approach to address them.

1.1 Motivation

As of this writing, the ideal object representation for robot grasping and manipulation tasks is yet unknown. The existing representations may not be the best for tackling more complex tasks as they lack actual object information belonging to the same class and configuration (shape, color and size).

In industrial robot-based automation, the objects are specifically coded for their visual features using 2D and 3D vision systems. The downside of this lies in the fact that the robot has to be taught to pick every other part with its visual representation. This process comes with the tedious schedule of teaching the robot to pick every part irrespective of the part's configuration, and viewpoint. The solution lies in using artificial intelligence powered robots.

With so much focus on modern artificial intelligence, it is easy to overlook that the topic is not new. Artificial intelligence has undergone several phases, each defined by whether the emphasis was on proving logical theorems or attempting to emulate the human mind through neurology.

Artificial intelligence may be traced back to the late 1940s when computer pioneers such as Alan Turing and John von Neumann began investigating how machines might "think." However, a watershed moment in AI happened in 1956, when researchers demonstrated that if given an infinite amount of memory, a machine could answer any issue.

The development of artificial intelligence made great strides in the 21st century. The first significant breakthrough was the creation of the self-learning neural network. By 2001, it had already outperformed humans in several domains, including object categorization and machine translation. Researchers enhanced its performance across a wide range of tasks over the following few years, thanks to advancements in the underlying technology. To a great extent artificial intelligence is empowered by machine learning and by Deep Learning (DL) in particular.

Machine learning enables computers to learn from data and experience in order to enhance their performance on certain tasks or decision-making processes. For this reason, machine learning employs statistics and probability theory. Machine learning employs algorithms to read data, learn from it, and make decisions without the need for explicit programming. Machine learning algorithms are frequently classified as either supervised or unsupervised. Supervised algorithms may apply previous learning to new data sets, whereas unsupervised algorithms can derive conclusions from datasets. Machine learning algorithms are programmed to seek out linear and non-linear correlations in a set of data. The learning is accomplished by the application of statistical approaches to

1 Introduction

train the algorithm to categorize or predict from a dataset.

DL extends machine learning using multi-layered artificial neural networks, i.e., the so-called Deep Neural Network (DNN) to achieve cutting-edge accuracy in object detection, speech recognition, and language translation. DNN is a critical technology underlying autonomous automobiles because it allows machines to analyze enormous volumes of complicated data, such as identifying people's faces in an image or video.

Artificial neural networks are based on biological neurons in the human brain and are made up of layers of linked nodes called "neurons" that include mathematical functions to analyze incoming input and anticipate an output value. Similar to how we learn from our parents, instructors, and peers, artificial neural networks learn by example. Deep learning models' performance continues to increase as more data is added.

A Convolutional Neural Network (CNN) is a DNN that is designed to pick up features and patterns in structural data such as images. For example, having an extensive collection of object photos even with more information such as depth and camera poses, CNN will easily learn the task-specific visual features. Furthermore, with task-specific optimizations, CNN can transform objects' visual features.

These CNN capabilities have enabled robots with skills to identify a part on an industrial fixture or in a bin since the networks have learned to recognise these parts from a series of images of these parts and other related visual data. If a new unseen part is in the bin, then the CNN is on the verge of object detection failure halting the robot cell in production or increasing the frequency of bin changing due to unpicked parts, which is an unfavorable industrial situation.

One of the ways to address this issue is by training a CNN to predict or regress a center point on an object silhouette which would then serve as a robot gripping point. This prediction is bound to fail when the object is of a complex or non-convex shape resulting in misalignment and subsequently in robot grasping failure. In such a case, the neural networks are additionally trained to predict the object pose. These joint predictions of object recognition and its pose are often computationally costly resulting in increase of the robot system engineering costs to meet the production goals.

We are now coming to an understanding that the two significant problems involve robot teaching for all the objects and in which orientation the robot has to pick the object.

Florence *et al.* [7] introduced a novel visual object representation to the robotics community, terming it 'dense object descriptors'. The dense object descriptors could generalize an object up to a certain extent and were recently applied to rope manipulation [8], block

1 Introduction

manipulation [9], robot control [10], fabric manipulation [11] and robot grasp pose estimation [12]. This prior research lays one of the solid foundations for this thesis for object generalization.

The next area of this thesis is to encompass the dense object descriptors-based representations with 6D-Pose¹ information about the object. Given a particular photograph of an object with its depth map (together RGB-D data) and its camera relative position, the photograph pixels can be converted to a set of 3D points or pointcloud [13]. The object pointcloud can be sampled using various methodologies [14, 15]. The obtained object pointcloud can be transformed into a 6D-Pose using Principal Component Analysis (PCA) [16] or Singular Value Decomposition (SVD) [17] methods. Meanwhile, DNN can be trained for multiple tasks [18]. We can, thus, adopt PCA or SVD methodologies to train the neural network to predict dense object descriptors with object poses while the DNN learns and embed both the task information.

If the object is occluded, then the pose associated with it changes causing robot grasping failure. Suwajanakorn *et al.* [19] introduced a method to train CNN for pose reconstruction based on geometrically consistent keypoints, i.e., a point that carries certain (predefined) properties. One of the advantages is that when the object is occluded, the network can recall the hidden keypoints in the occlusion. This way, such a network is still able to reconstruct object pose from which one can now compute robot grasping pose consistently.

¹6D-Pose of an object refers to the object position and rotation in a 3D space.

2 State of the Art

The term “state of the art” refers to the most advanced degree of development accomplished in a design, method, material, or technology. It is an important consideration in any engineering feat. The overview section focuses on recent developments in the machine vision field that generalize object representations for robot grasping.

2.1 Overview

In 2017, Schmidt *et al.* [20] introduced instance-level generalization of objects using dense object descriptors. The dense object descriptors are embeddings for pixels in an image such that all pixels are uniquely defined with respect to each other. Schmidt *et al.* [20] could generalize a human body pose with sequenced RGB-D data with mutual temporal correspondence using Scale-Invariant Feature Transform (SIFT) [21]. In this case, the dense object descriptors are computed using non-linear functions ($f : I_{RGB}[u, v] \in \mathbb{R}^3 \rightarrow I_D[u, v] \in \mathbb{R}^D$) converting an image pixel to an arbitrary vector length $D \in \mathbb{N}^+$. Based on this non-linear function as well as on the “Pixelwise Contrastive Loss” [22], Florence *et al.* [7] introduced DON which is a self-supervised network that converts an RGB picture into a descriptor space image that implicitly stores essential object features invariant to the viewpoint, configuration and illumination. DON’s self-supervised training is impressive. We can apply it to random objects and deploy it in mere in hours [7].

However, when employing consumer-grade cameras, the training strategy relies on correct depth photos and has difficulties with tiny, reflecting objects, which are common in industrial environments. In the meantime, Kupcsik *et al.* [12] used Laplacian Eigenmaps [23] to embed a 3D object model into an optimally generated space bringing in more domain knowledge by associating 3D model views to images views only. We can efficiently apply it to smaller texture-less and reflective objects as it does not rely on depth information, thus, overcoming the aforementioned drawback of DON. [7] Kupcsik *et al.* [12] further compare training strategies for producing 6D grasps for industrial objects and show that a unique supervised training approach increases pick-and-place resilience in industry-relevant tasks.

Differently, Hadjivelichkov and Kanoulas [24] extends the DON to address the problem of inferring self-supervised dense semantic correspondences between objects in multi-object or cluttered scenes. Their technique introduces the learning of class-aware dense

object descriptors by providing either confidence in object similarities or unsupervised discrete labels.

Florence [25] has found that the “Pixelwise Contrastive Loss” function used in DON might not perform well if a computed correspondence is wrong, and the precision of contrastive-trained models can be sensitive to the relative weighting between positive-negative sampled pixels. Instead, the authors introduce a new continuous sampling-based loss function that is much different from the space sampling-based “Pixelwise Contrastive Loss” function called “Pixelwise Distribution Loss”, which regresses a set of probability distributions heatmaps aiming to minimize the divergence between the predicted heatmap and the ground truth heatmap mitigating errors in human data labeling errors and depth measurements noise by consumer-grade cameras. Based on [26, 27], Adrian *et al.* [28] introduced another novel loss function called “Pixelwise NT-Xent loss” to train DON more robustly. Adrian *et al.*’s experiments show that the novel loss function is invariant with respect to the batch size and the number of correspondences and used “PCK@ k ” metric to further evaluate and benchmark DON on cluttered scenes previously not benchmarked.

Concerning object poses, Manuelli *et al.* [29] offer an innovative formulation of category-level objects that are represented by semantic 3D keypoints with the manipulation specified by geometric costs and restrictions on those keypoints. The formulation naturally enables the manipulation strategy to encompass 3D keypoint recognition, optimization-based robot action planning, and grasping-based action execution. Vecerik *et al.* [30] further introduces self-supervised techniques in keypoint prediction for robot manipulability based on [31, 32, 33, 34], significantly reducing the need for human-labeled data and providing a robust detector for semantic 3D keypoints in their setup ‘SEK’. Its main contribution comes from considering multi-view geometry as a source of self-supervision for keypoint-based models. Furthermore, Vecerik *et al.* [30] show the applicability of SEK to robotic tasks. A small set of labelled examples shows other performance scales with the total amount of data, not the total labelled data. It further shows how we can use unlabeled data combined with “SEK” to counteract the effects of domain shift and its ability to generalize across samples from the same category.

Suwajanakorn *et al.* [19] put forth geometrically consistent keypoint nets, exploring the idea of optimizing a representation based on a sparse collection of keypoints or landmarks, but without access to keypoint annotations, using an end-to-end geometric reasoning framework. The paper shows that using two unique objective functions, namely, a relative pose estimation loss and a multi-view consistency goal, uncovers the consistent keypoints across multiple views and object instances. Their affine translation-equivariant design may extend to previously unknown object instances and ShapeNet [35] categories. The identified keypoints on stiff 3D pose estimation surpass those from

a directly supervised learning baseline.

2.2 Breakdown Analysis & Comparison of Influential SOTA

In this section, the SOTA influential to this thesis is compared for its contribution, drawbacks with respective to the thesis addressed as relative drawbacks.

Contributions	Relative Drawbacks
Self-supervised Visual Descriptor Learning for Dense Correspondence[20]	<ul style="list-style-type: none"> Trained a CNN trained on contrastive loss. Produced generalized viewpoint & lighting invariant descriptors.
Dense Object Nets: Learning Dense Visual Object Descriptors By and For Robotic Manipulation[7]	<ul style="list-style-type: none"> Using SIFT for correspondence mapping isn't spatially accurate.
<ul style="list-style-type: none"> Used camera-based transformation to map correspondence in image pairs to train DON. Quick trainable DON. Formulated "Pixelwise Contrastive Loss" to train the network. 	<ul style="list-style-type: none"> The loss function is dependent on number of correspondences. The network performance degrades while training for batch size > 1. The descriptors have non-optimal spatial expectations when queried.
Dense Visual Learning for Robot Manipulation[25]	
<ul style="list-style-type: none"> Introduced "Pixelwise Distribution Loss" robust against incorrect correspondence sampling encountered by pixelwise contrastive loss. Does not need non-matches to train DON. 	<ul style="list-style-type: none"> The pixelwise distributional loss function cannot accommodate same objects in an image while training.
Supervised Training of Dense Object Nets using Optimal Descriptors for Industrial Robotic Applications[12]	

Continued on next page

Breakdown Analysis & Comparison of Influential SOTA – continued

Contributions	Relative Drawbacks
<ul style="list-style-type: none"> Emphasized that DON's are not efficient while training on small & shiny objects due to the consumer grade depth cameras. Used 3D model with Laplacian Eigenmaps for supervised training of DON. 	<ul style="list-style-type: none"> Used 3D models to remap objects in images for reconstructing depths. Laplacian Eigenmaps are computational costly and do not handle occlusion.
NeRF-Supervision: Learning Dense Object Descriptors from Neural Radiance Fields[36]	
<ul style="list-style-type: none"> Used Neural Radiance Fields (NeRF) to train DON making it robust against shiny surfaces. 	<ul style="list-style-type: none"> NeRF does not work in multi-object scenes where an object is missing in another viewpoint.
Fully Self-Supervised Class Awareness in Dense Object Descriptors[24]	<ul style="list-style-type: none"> Introduced loss functions to train DON in an unsupervised manner. The loss functions introduced hard and soft margins to pixelwise contrastive loss.
Efficient and Robust Training of Dense Object Nets for Multi-Object Robot Manipulation[28]	<ul style="list-style-type: none"> Does not handle occluded scenes.

Continued on next page

Breakdown Analysis & Comparison of Influential SOTA – continued

Contributions	Relative Drawbacks
<ul style="list-style-type: none"> • Adapted NTXent-Loss on a pixel-wise scale to train multiobject scene DON. • Proposed that DON relies on color hues of objects to reconstruct geometric priors. • Benchmarked all DON loss functions. • Used synthetic correspondences to train DON without depth information. 	<ul style="list-style-type: none"> • Did not accomodate training DON that belongs to the same class.
kPAM: Keypoint Affordances for Category-Level Robotic Manipulation[29]	
<ul style="list-style-type: none"> • Introduced object representation using 3D-Keypoints to generalize pose. • Avoids task-inappropriate geometric information. 	
S3K: Self-Supervised Semantic Keypoints for Robotic Manipulation via Multi-View Consistency[30]	
<ul style="list-style-type: none"> • Improved object representation with semantic 3D-keypoints. • Used multi-view consistency loss for training robustly against occlusion, noise and absence of visible texture. 	
Discovery of Latent 3D Keypoints via End-to-end Geometric Reasoning[19]	

Continued on next page

Breakdown Analysis & Comparison of Influential SOTA – continued

Contributions	Relative Drawbacks
<ul style="list-style-type: none"> • Introduced ‘KeypointNet’ for geometric reasoning. • Used pretrained orientation network for making predictions on symmetrical objects more robust. • Used self-supervised loss functions with end-to-end training methods. 	<ul style="list-style-type: none"> • Uses chordal distances for computing distances in rotational matrices. • Accommodates objects belonging to the same class.
<p>Neural Descriptor Fields: SE(3)-Equivariant Object Representations for Manipulation[37]</p> <ul style="list-style-type: none"> • Used neural descriptor model to fuse models of DON, Kpam[29] and NeRF for object manipulation. • Used neural energy fields to find generalized keypoints for manipulation. • Used self-supervised loss functions with end-to-end training methods. 	<ul style="list-style-type: none"> • Need robot demonstrations to train the neural networks.

Table 2.1: Comparison of influential SOTA.

3 Objective

This thesis focuses on engineering a single semantic class-generalized label ($x \in \mathbb{R}^D$) acting as a robot grasping point in an image. If no possible labels are found in the image search space then, “KeypointNet” is deployed as a fail-safe to regress geometrically consistent keypoints on the object in turn generating a valid 6D-Pose of the object for robot grasping.

As the recent research proceedings [7, 28, 8, 9] introduced different CNN-based Residual Network (ResNet) [38] architectures and loss functions to implement DON, the best combination of ResNet architecture and loss function will be chosen by benchmarking all the valid combinations.

Furthermore, one object will be selected for generalization using DON to manually store a valid specific generalized label for robot grasping. To eliminate the process of manually selecting labels, ‘KeypointNet’ will be implemented.

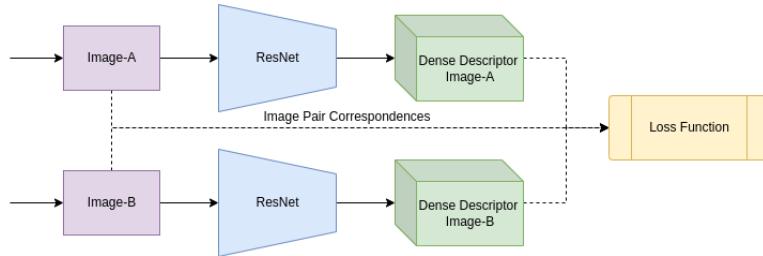
Based on the availability of the datasets, synthetic data [39] will be generated to train the neural networks if no ready datasets are available.

4 Methodology

4.1 Dense Object Nets

DON is a recently proposed self-supervised contrastive DNN framework. The network architecture is based on adaptation of ‘SIMCLR’[40] architecture refer Figure 4.1. While the ‘SIMCLR’ architecture encodes the entire image to an embedding ($f_{SIMCLR} : I \in \mathbb{R}^{H \times W \times 3} \rightarrow x \in \mathbb{R}^D$) using the contrastive loss function the DON, encodes each pixel in an image to embedding space called dense descriptors refer equation 4.1. The image with dense descriptors as pixels is referred as dense descriptor image.

Figure 4.1: DON architecture.



Furthermore, according to Adrian *et al.* [28], DON relies on sequence of images in the dataset to embed each pixel to descriptor space by exploiting the geometric prior of objects in turn, generalizing the object.

$$f_{DON} : I \in \mathbb{R}^{H \times W \times 3} \rightarrow I_D \in \mathbb{R}^{H \times W \times D} \quad , \text{s.t. } D \in \mathbb{N}^+. \quad (4.1)$$

The ResNet is usually optimized using mutual image correspondence information of image pairs with contrastive learning on a pixel scale. The trained network computes embeddings such that (s.t.) the pixel has an unique embedding compared to the rest of the pixels. The network’s visual embedding is invariant to a viewpoint and illumination and comes with the inherent property of generalizing same-class objects. The visual embedding uniquely describes the pixel locally in an image; hence, referred to as a local descriptor. The contribution of DON is thus its immense ability to generalize for other similar objects within a class without a need to train the network for each such object.

4.1.1 Implementation Strategy

Introduced in 2019, DON is young in the research field and applied for use cases in 2021. In earlier years, the authors of DON proposed “Pixelwise Distribution loss” to improve “Pixelwise Contrastive loss”. This year, researchers Adrian *et al.* [28] introduced a new loss function called “Pixelwise NT-Xent Loss” to make DON robust. The implemented use cases [8, 9, 10] used different ResNet architecture variants giving an opportunity to test combinations of loss functions with architectures to optimize the network.

4.1.2 ResNet Architectures

ResNet- N means that the DNN has N convolution layers with shortcuts between them aiding better optimization [38]. E.g., ResNet-34 consists of 34 weighted layers. Originally proposed, ResNet-34 came with few design rules; to maintain the temporal complexity per layer, the layers had the same number of filters for the same output feature map size, and the number of filters doubled in that scenario. The identity shortcuts were employed directly as the input and output dimensions stand the same.

With an increase in dimensions, there were two options to consider. The initial assumption was that the shortcut would continue to conduct identity mapping while padding extra zero entries for expanding dimensions. The other option was to utilize the projection shortcut to match dimensions.

The ResNet-50 inherits architecture from the ResNet-34 model, with revision in the bottleneck design. Each of the ResNet-34’s 2-layer blocks was replaced with a 3-layer bottleneck block, resulting in the deeper ResNet-50 design.

4.1.3 Loss Functions

Pixelwise Contrastive Loss

The pixelwise contrastive loss is adapted from [40] on a pixelwise scale. The pixelwise contrastive loss brought in the concept of DON itself was introduced by Florence *et al.* [7]. The pixelwise contrastive loss function aims to minimize the distance between the embeddings corresponding to a match, while the embeddings corresponding to a non-match are set at least by a marginal distance M apart. The loss function is described as,

$$\mathcal{L}_{matches} = \frac{1}{N_{matches}} \sum_{i,j}^N \|f(I_a)[x_i, y_j] - f(I_b)[x_i, y_j]\|, \quad (4.2)$$

$$\mathcal{L}_{non-matches} = \frac{1}{N_{non-matches}} \sum_{i,j}^N \max(0, M - \|f(I_a)[x_i, y_j] - f(I_b)[x_i, y_j]\|), \quad (4.3)$$

$$\mathcal{L}_{Total} = \mathcal{L}_{matches} + \mathcal{L}_{non-matches}. \quad (4.4)$$

M is a hyperparameter and has to be chosen optimally while training the network. The loss function is sensitive to batch size and number of correspondences [28, 25]. Additionally, non-matching correspondences needs to be sampled for training the network. This thesis is not going to benchmark this loss function as it is outdated and comes with drawbacks compared to newly proposed loss functions.

Pixelwise Distribution Loss

Introduced in Florence *et al.* [7], the pixelwise distribution loss function aims to compute a set of probability distributions for each keypoint $k = \{1, 2, \dots, K\}$ minimizing the divergence between the predicted heatmap and the ground truth heatmap described as,

$$p_a(x_i, y_i | f(I_a), f(I_b), x_b, y_b) = \frac{K(f(I_b)[x_b, y_b], f(I_a)[x_i, y_i])}{\sum_{i', j'} K(f(I_b)[x_b, y_b], f(I_a)[x'_i, y'_j])}, \quad (4.5)$$

where, the $K : (\mathbb{R}^{\mathbb{N}^+}, \mathbb{R}^{\mathbb{N}^+}) \rightarrow \mathbb{R}$ is a kernel function and it can be of any formulation as long as it yields a scalar (\mathbb{R}) processing two vectors of the same dimension. The term, $p_a(x_i, y_i | f(I_a), f(I_b), x_b, y_b) \in [0, 1]$ is the spatial probability of a keypoint in $f(I_a)$ in the image $f(I_b)$.

The spatial expectation from the spatial probability is computed as,

$$\hat{x}_b = \sum_i^W p_a(x_i | f(I_a), f(I_b), x_b) x_i, \quad (4.6)$$

$$\hat{y}_b = \sum_i^H p_a(y_i | f(I_a), f(I_b), y_b) y_i, \quad (4.7)$$

The spatial loss of $I_a[x_a, y_a]$ in $I_b[x_b, y_b]$ is computed as,

$$\mathcal{L}_{b \rightarrow a} = \|x_b - \hat{x}_b\| + \|y_b - \hat{y}_b\|, \quad (4.8)$$

The total bi-directional loss is computed as,

$$\mathcal{L}_{Total} = \mathcal{L}_{b \rightarrow a} + \mathcal{L}_{a \rightarrow b}, \quad (4.9)$$

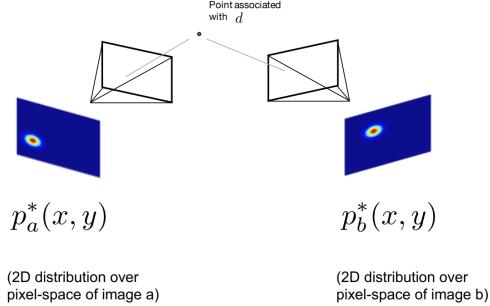
The kernel functional chosen to compute the bi-directional loss function,

$$K : (f(I_a)[x_i, y_i], f(I_b)[x_b, y_b]) \rightarrow \exp(-\|f(I_a)[x_i, y_i] - f(I_b)[x_b, y_b]\|). \quad (4.10)$$

This pixelwise distribution loss function can be implemented in a batch fashion yielding high-performance computation time in a computer. The Figure 4.2 intuitively represents

Figure 4.2: 2D spatial expectation of keypoint in two images.

Source: [25]



the spatial expectation of a keypoint.

Furthermore, the loss function improves spatial expectation of a descriptor compared to the pixelwise contrastive loss, yielding better results in the search space of a descriptor in a new image. One of the drawbacks of using this loss function is that only one object belonging to the same class in an image can be used while training the neural network. Having more objects belonging to the same class yields faulty prediction of spatial expectations.

Pixelwise NT-Xent Loss

The “NT-Xent loss” is originally introduced in Chen *et al.* [40] adopted by Oord *et al.* [41] and implemented on a pixel level to train DON by Adrian *et al.* [28]. The authors Adrian *et al.* [28] randomly sample N correspondences from an image pair (I_A, I_B) , with each correspondence yielding two descriptors d_i^A and d_i^B , for a total of N pairs, or $2N$ descriptors. For a given descriptor, all other $2(N - 1)$ descriptors are respectively treated as negative examples. The below given loss function is directly optimized against the descriptor pair,

$$\mathcal{L}_{i,j} = -\log \frac{\exp(K(d_i, d_j)/\tau)}{\sum_{k=1, k \neq i}^{2N} \exp(K(d_i, d_k)/\tau)}, \quad (4.11)$$

where, $\tau \in \mathbb{R}^+$ is a temperature scaling factor and K is cosine similarity kernel defined as,

$$K = \frac{\langle d_i, d_j \rangle}{\|d_i\| \|d_j\|}, \quad \text{s.t. } d_{i,j} \in \mathbb{R}^{\mathbb{N}^+}, K \in [1, 0]. \quad (4.12)$$

Since K takes only positive inputs, unlike the cosine function, it produces the outputs in the range $[0, 1]$ which leads to its interpretation as in expression refer equation 4.13.

$$K(d_i, d_j) = \begin{cases} 1 & \Rightarrow (d_i, d_j) \text{ are similar.} \\ 0 & \Rightarrow (d_i, d_j) \text{ are dissimilar.} \end{cases} \quad (4.13)$$

One of the drawbacks of the pixelwise nt-xent loss function is that semantically closer objects in an image are treated as two different objects which in turn hinders the generalizing properties of the neural networks.

4.1.4 PCK@k as an Evaluation Metric

To measure the performance of the network, “PCK@k” evaluation metric is adopted as in preceding works [42, 28]. PCK@k is defined for a set of pixel pairs $\mathcal{T} = \{p_i, q_i\}_{i=1}^N$ as follows,

$$PCK@k(\mathcal{T}) = \frac{1}{|\mathcal{T}|} \sum_{(p_i, q_i) \in \mathcal{T}} [\|p_i - q_i\| \leq k] \quad , \text{s.t. } (p_i, q_i) \in \mathbb{R}^{N^+}, (k, \mathcal{T}) \in \mathbb{R}, \quad (4.14)$$

where k is the maximum error allowed in turn, influencing the Iverson bracket $[\cdot]$ to 1 if the error is within the limits of k and 0 otherwise. \mathcal{T} is a number of keypoints (p_i, q_i) normalizing the computed metric. Instead of using L^2 -Norm to compute the error between keypoints, it is replaced by a kernel function ‘ D ’ similarly in equation 4.12 in page 15 bringing in more flexibility. The kernel revised metric is described as,

$$PCK@k(\mathcal{T}) = \frac{1}{|\mathcal{T}|} \sum_{(p_i, q_i) \in \mathcal{T}} D(p_i, q_i) \leq k. \quad (4.15)$$

4.1.5 Suzanne, a synthetic dataset

To train and benchmark DON, no compatible ready datasets are available. The datasets available either missed camera information like intrinsics or extrinsics parameters, masks information or captured same objects belonging to the same class in an image, posing ill-conditions to benchmark pixelwise distribution loss.

To overcome this situation, ‘BlenderProc’ [43] is used to generate synthetic dataset with all annotations needed to train DON. ‘Suzanne’ is a model already available in the render tool and is placed in the world datum center. For this control dataset, 100 random cameras are added in the environment with random poses capturing depth, extrinsic information and mask for each viewpoint. The Table 4.1 in page 17, provides the preview of dataset. The annotated pixels in blue in second image from right to left in Table 4.1 projects the correspondences through world coordinates in the first image from right to left annotated in red. As the dataset produces perfect depth maps of the objects in the images, the benchmarking metrics might not tally with available benchmarked results.

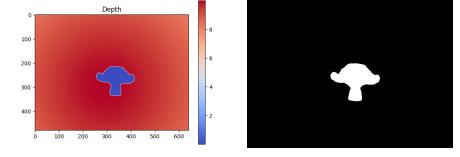
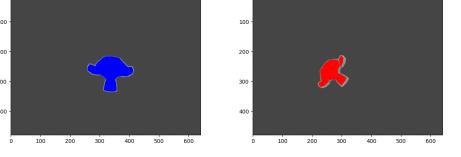
Dataset Preview			
RGB	Depth	Mask	Correspondences Mapping
			

Table 4.1: Depiction of dataset quintennials features of dataset

The camera extrinsic parameters ($\mathcal{T}_{C \rightarrow W} \in \mathbb{R}^{4 \times 4}$) is the camera pose relative to the world frame. The camera intrinsics is defined as $\mathcal{I} \in \mathbb{R}^{3 \times 3}$. To map the correspondences from image I_a to image I_b , the pixels from the image $[u, v]^T \in I_a \in \mathbb{R}^{2 \times 1}$ are projected to the world coordinates $W_a \in \mathbb{R}^{3 \times 1}$ using depth (d) and from the world coordinates to the image I_b as follows:

$$W_a = \mathcal{T}_{C \rightarrow W}^a \mathcal{I}^{-a} \begin{bmatrix} u^a \\ v^a \end{bmatrix} d_{uv}^a \quad (4.16)$$

The computed world coordinates from I_a , W_a are projected to pixels in I_b ,

$$d_{uv}^b \begin{bmatrix} u^b \\ v^b \end{bmatrix} = \mathcal{I}^b \mathcal{T}_{C \rightarrow W}^{-b} W_a, \quad (4.17)$$

where,

$$\mathcal{T}_{C \rightarrow W}^{-b} = \begin{bmatrix} R^b & -R^b t^b \\ 0_3^T & 1 \end{bmatrix}, \quad \text{s.t. } R^b \in SO(3). \quad (4.18)$$

Additionally, mask is used to sample the pixels from I_a within the object silhouette.

4.1.6 Manual Method for Robot Grasp Generation

The DON produces visual embeddings ($f(I)[u, v] \rightarrow d, d \in \mathbb{R}^{\mathbb{N}^+}$) such that each pixels withholds unique embedding in descriptor space reducing computational complexity to find the same pixel in different viewpoint in an 3-Channel Image (RGB) space omitting complex search algorithm. Using “Pygame” [44] library in python, an interactive application is developed to pick descriptors in descriptor space with its image coordinates $[u, v]^T$ using depth and camera intrinsics to project them to the coordinates in the camera frame. More than three descriptors are picked such that it produces a valid rotation matrix via PCA or SVD and the mean value acting as a center to generate a 6D pose defined as,

$$\mathcal{T} = \begin{bmatrix} R & t \\ 0_3^T & 1 \end{bmatrix} \in \mathbb{R}^{4 \times 4}, \quad \text{s.t. } R \in SO(3), t \in \mathbb{R}^{3 \times 1}, \quad (4.19)$$

The interactive application allows to save multiple object pose definition file and this definition file is loaded into another application to search and compute multiple object 6D pose information in another or same image. The descriptors are searched using gaussian kernel derived from heat kernel formulation,

$$\mathbb{E}[u^*, v^*]_d = \underset{u_i, v_i}{\operatorname{argmin}} \exp - \left(\frac{\|(f(I_a)[u_i, v_i] - d)\|}{\exp(t)} \right)^2 \quad (4.20)$$

Where $t \in \mathbb{R}$ controls the kernel width influencing the search space to compute the spatial expectation of the query descriptor d in image I_a . The descriptors produced by DON are supposedly to be robust against color jitters, illumination changes and occlusions in order to produce consistent 6D poses [28].

To compute the relative 6D pose of an object in the new viewpoint of the camera, the queried keypoint locations are reordered in the matrix respective to the keypoint and its spatial locations as stored in the definition file. If a keypoint is not found, then the keypoint in the definition file is omitted and mutually rearranged. The queried keypoint locations produce relative 6D pose $\mathcal{T}_{\text{object}}$ compared to the pose stored in the definition file as,

$$\mathcal{T}_{\text{rel}} = (\mathcal{T}_{\text{def}}^T \mathcal{T}_{\text{def}})^{-1} \mathcal{T}_{\text{def}}^T \mathcal{T}_{\text{object}}. \quad (4.21)$$

To eliminate the manual process of annotating keypoints to generate robot grasps, KeypointNet is implemented.

4.2 KeypointNet

KeypointNet is a framework for end-to-end geometric reasoning that regresses the best set of category-specific 3D keypoints with their associated detectors, i.e., higher-dimensional generalized embeddings. By definition, KeypointNet works for an image pair. This framework estimates 3D pose by giving a differentiable goal that seeks the optimal set of keypoints to recover the relative pose of an item between two viewpoints. Hence, KeypointNet allows for predicting geometrically and semantically consistent keypoints across viewing angles and instances of an object category by optimizing four loss functions.

4.2.1 Loss functions

The KeypointNet produces N number of spatial probabilities for N keypoints described by $g_i(u, v) \in \mathbb{R}^{W \times H}$, s.t. $\sum_{u,v} g(u, v) = 1$. The pixel spatial expectations is computed via continuous sampling method described by,

$$[u_i, v_i]^T = \left[\sum_{u,v} g_i(u, v) * u, \sum_{u,v} g_i(u, v) * v \right]^T, \quad \text{s.t. } g_i(u, v) \in \mathbb{R}^{W \times H}. \quad (4.22)$$

Similarly, depth is computed as follows:

$$d_i = \sum_{u,v} g_i(u, v) * d_i[u, v], \quad \text{s.t. } d_i(u, v) \in \mathbb{R}^{W \times H}. \quad (4.23)$$

The self-supervised training of this network is based on the following loss functions.

Multiview Consistency Loss

The multiview consistency loss function enforces the network to regress keypoint expectations at same the locations in an image pair. Concisely, a keypoint in one image should project on a pixel location corresponding to a keypoint in another image. To achieve this, pixels are projected to the camera frame using depth and intrinsic information. Using mutual transformation information between two cameras, the pixels are further projected back to the pixel frame of the second camera. The loss is then the distance between the projected pixel location and regressed keypoint's pixel location.

To make this formulation more intuitive, the originally proposed method is altered. Instead, of evaluating the keypoints in pixel frame, they are evaluated in the world frame. The ideology is that the object is fixed in the world datum and camera poses changes preserving the object coordinates in the world frame refer Figure 4.2 in page 15. The altered multiview consistency loss function is defined as,

$$\mathcal{L}_{mvc} = \frac{1}{N} \sum_{i=1}^N \|W_A^i - W_B^i\|, \quad \text{s.t. } W_{A,B} \in \mathbb{R}^{3 \times 1}. \quad (4.24)$$

Relative Pose Loss

As the KeypointNet produces N number of keypoints, the keypoint locations together projected into camera frame coordinates produces a 6D pose computed via SVD. The relative pose loss functions rearranges the keypoint locations in an image preserving the object's 6D pose in the viewpoint respective to the camera pose. Additionally, this preserves the relative pose between two camera poses and relative pose generated from the keypoint locations in an image pair.

As per the original implementation of the KeypointNet, the authors discard the relative translation and expressed the relative pose loss in form of chordal distance between two rotational matrices $\in SO(3)$ using Frobenius-Norm [45] as:

$$\mathcal{L}_{pose} = 2 \arcsin \left(\frac{1}{2\sqrt{2}} \|\hat{R} - R\|_F \right), \quad \text{s.t. } R \in SO(3). \quad (4.25)$$

4 Methodology

The Frobenius norm yields the chordal distance in range $[0, 2\sqrt{2}]$ which is normalized in the implementation. Satisfying the triangle inequality condition, this function “*some messy algebra*” [46] as the operation $\hat{R} - R \notin SO(3)$.

To make this more geometrically intuitive, the poses $\in SE(3)$ are transported to *Riemannian manifold* [47]. *Riemannian manifold* is rich in geometry and differentiably smooth. Instead, of expressing the distance in chordal form, ‘geodesic-distance’ is chosen as it is smooth on for infinitesimally small trajectories on the manifold. Here, the trajectory is defined as the distance to be covered such that the two rotation matrices align with each other.

The manifold geometry consists of two parts, i.e., a manifold and a tangential plane tangential place $T(\mathcal{M}^p)$ represented as a tangential Euclidean space to the manifold, and the manifold space. the manifold space $\mathcal{M}^p : p \in \mathbb{R}^{N^+}$. Exp-maps and log-maps refer equation 4.27 are used to transport a vector on tangential plane to the manifold and vice versa.

$$expmap : \mathfrak{so}(3) \rightarrow SO(3), \quad (4.26)$$

$$logmap : SO(3) \rightarrow \mathfrak{so}(3). \quad (4.27)$$

Here $r(\alpha) \in \mathfrak{so}(3) \in T(\mathcal{M}^p)$ is a skew-symmetric matrix built from the axis angles $\alpha \in \mathbb{R}^3$ described as,

$$r(\alpha) = \begin{bmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{bmatrix}, \text{s.t. } \alpha = [x, y, z]^T \in \mathbb{R}^3. \quad (4.28)$$

The object or camera’s rotation in the dataset is represented as $SO(3)$.

The geodesic distance is a *bi-invariant* metric on $SO(3)$ with range $[0, \pi]$:

$$\mathcal{L}_{pose} = \frac{1}{\sqrt{2}} \|logmap(R_A^T R_B)\|_F = \frac{1}{2} \|logmap(R_A^T R_B)\|. \quad (4.29)$$

The \mathcal{L}_{pose} is substituted in the place of original loss function formulation. The relative groundtruth rotation (\hat{R}) for the loss is computed as:

$$\hat{R}_{A \rightarrow B} = \hat{R}_A^{-1} \hat{R}_B, \text{s.t. } \hat{R} \in SO(3). \quad (4.30)$$

$\hat{R}_{A \rightarrow B}$ is substituted in the equation 4.29 with R_A , and in order to compute the predicted rotation R_B created from the set of predict points $X_{A,B} \in \mathbb{R}^{N \times 3}$ via Kabsch’s algorithm [48], SVD of covariance matrix \mathcal{H} is used as follows,

4 Methodology

$$\mathcal{H} = (X_A - X_A^{mean})^T (X_B - X_B^{mean}) \quad , \text{s.t. } \mathcal{H} \in \mathbb{R}^{3 \times 3}, X_{A,B}^{mean} = \bar{X}_{A,B}. \quad (4.31)$$

where, $(X - X^{mean})$ centers the X to origin with the mean $X^{mean} \in \mathbb{R}^{1 \times 3}$ computed along the dimension N where each row is subtracted with the mean. The optimal rotation is computed as,

$$R_B = V \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \det(VU^T) \end{bmatrix} U^T, \quad (4.32)$$

where,

$$U\Sigma V^T = SVD(\mathcal{H}). \quad (4.33)$$

The term $\det(VU^T)$ handles the reflection case while estimating the rotation matrix. In other terms it makes the z rotation vector orthonormal to ' xy ' rotation vectors such that it always points to the camera in the camera relative frame. This method to condition z rotation vector produces gradients which allows neural networks to be trained via backpropagation compared to using cross-product of vectors which does not produce gradients since the vector in the matrix is changed via inplace operation.

Separation Loss

Separation loss function penalizes cases where the distance between two keypoint locations is less than a specified margin (δ).

$$\mathcal{L}_{sep} = \frac{1}{N} \sum_i^N \sum_{j \neq i}^N \max(0, \delta^2 - \|X_i - X_j\|^2). \quad (4.34)$$

Silhouette Loss

Silhouette loss functions encourages the spatial probabilities of keypoints $g_i(u, v)$ to lie with the object silhouette:

$$\mathcal{L}_{obj} = \frac{1}{N} \sum_{i=1}^N -\log \sum_{u,v} b[u, v] g_i(u, v). \quad (4.35)$$

While training the network, the mask of the object is accessible. The mask values are sampled as $b[u, v] \in \{0, 1\}$. The mask information is used only while training the network and is not used during the inference. If all the keypoints are regressed within the object silhouette, the loss computes to zero.

Variance Loss

Variance loss function penalizes the spatial probabilities if they are broad in turn, encouraging peaky distribution. The broader spatial probabilities brings in more uncertainty in the keypoint spatial expectation. In turn, the loss function penalizes the uncertainty of the keypoint spatial expectation by shrinking the span of the spatial probabilities as depicted in Figure 4.3 in page 22. Furthermore, in the Figure 4.3 in page 22, the x marks the keypoint spatial expectation. The left image illustrates the broader spatial probability of a keypoint bringing in more uncertainty in the keypoint spatial location (expectation). The right image describes the peaky spatial probability of the keypoint encouraged by the loss function.

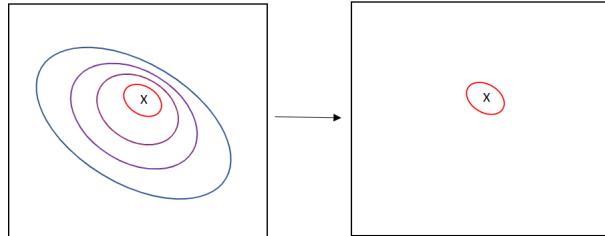
The variance loss functions is described as,

$$\mathcal{L}_{var} = \frac{1}{N} \sum_{i=1}^N \sum_{u,v} g_i(u,v) \left\| \mathcal{G} - [u_i, v_i]^T \right\|^2, \quad (4.36)$$

where \mathcal{G} is the spatial grid of row and column indices described as,

$$\mathcal{G}_{0ij} = \begin{bmatrix} 0 & \dots & 0 \\ \vdots & \ddots & \vdots \\ H & \dots & H \end{bmatrix}, \quad \mathcal{G}_{1ij} = \begin{bmatrix} 0 & \dots & W \\ \vdots & \ddots & \vdots \\ 0 & \dots & W \end{bmatrix} \quad , \text{s.t. } \mathcal{G} \in \mathbb{R}^{2 \times W \times H}. \quad (4.37)$$

Figure 4.3: Variance loss function in work.



4.2.2 Total Weighted-Sum Loss

To train the network, all loss functions are weighed by their own specific weights in turn, signifying that one loss functions needs more priority than other loss functions. The priority depends on the network architecture being used. Let's say if the multiview consistency loss is not converging, then the weights assigned to that particular loss function are increased such that the task is fulfilled by the network. The loss function is the scalar product of the weights $W = [w_{mvc}, w_{pose}, w_{sep}, w_{obj}, w_{var}]^T$ and the individual loss functions $\mathcal{L} = [\mathcal{L}_{mvc}, \mathcal{L}_{pose}, \mathcal{L}_{sep}, \mathcal{L}_{obj}, \mathcal{L}_{var}]^T$ described as,

$$\mathcal{L}_{Total} = \langle W, L \rangle , \text{s.t. } \mathcal{L}_{Total} \in \mathbb{R} \text{ & } W, L \in \mathbb{R}^4. \quad (4.38)$$

Here W is a hyperparameter. The hyperparameter in this thesis implementation will be fixed by a trial-and-error process as other sampling or optimization process consumes a lot of time.

4.2.3 Network Modification

The ResNet architecture will be used over originally proposed image segmentation based CNN while preserving the part of architecture with predicting depth and spatial probabilities of the keypoints to preserve the neural network architecture homogeneity with DON architecture.

Additionally, no pretrained orientation network is used to aid the KeypointNet breaking the symmetry of the objects as in [19]. As per the proposed data augmentation in [19], the data will not be scaled to a unit dimension preserving realistic conditions of object in an environment for which the network will be trained for.

4.3 End-to-End Network Training

End-to-end network training is a deep learning technique in which all parameters are taught concurrently rather than step by step. A deep learning method is trained in an end-to-end manner when a DNN needs to perform multiple tasks. Furthermore, sometimes a deep learning network learns joint features pertaining multiple tasks. Particularly in this thesis, joint features for DON representations and generalized object 6D poses are computed via generalized geometrically consistent keypoints.

NOTE: Each sub-section described in this part of thesis qualifies for scientific publication.

4.3.1 KeypointNet Informed Dense Object Nets

The depth values produced for small & shiny objects by the consumer grade depth cameras are not accurate [12], in turn leading to inconsistent correspondences mapping between two image pairs which further leads to inaccurate training of DON. To overcome this, KeypointNet is employed.

As per the definition, KeypointNet can be modified to generate correspondences between image pair belonging to the same class without the need of additional DNN like Universal Correspondence Network [49]. Using KeypointNet eliminates the need for projecting pixels from one image to another image through the world coordinates Equation 4.16 in page 17, and this method can accommodate a single object and not similar

objects belonging to same class.

Furthermore, the KeypointNet architecture is stacked on top of DON architecture such that semantic correspondences are supplied for DON seamlessly keeping it self-supervised. Both the networks are optimized with the joint losses and not individually as described below:

$$\mathcal{L}_{Joint} = \mathcal{L}_{KeypointNet} + \mathcal{L}_{DON} \quad (4.39)$$

The joint losses produces mutual gradients for optimization such that the networks compute features which are beneficial to each other forming a coordination to fulfil a task.

Furthermore, DON training is delayed by few epochs and the KeypointNet is trained at the start such that the training converges faster.

4.3.2 Dense Object Nets Informed KeypointNet

Preserving the goal of the thesis, single class generalized labels need to be stored for deployment and to achieve this, DON is stacked on top of KeypointNet. As DON produces class generalized descriptors in form of labels the KeypointNet is employed to pick the labels autonomously.

The KeypointNet regresses N number of keypoints in the descriptor image and each keypoint is a pixel location for the label in the descriptor image. For an object, class generalized keypoints are stored in N random probabilities. This insures that in cases of object occlusion in viewpoint, if one label is not found, another label can be recalled creating a grasping point for the robot as a counter-measure.

For better optimization, the DON is trained first and few epochs later KeypointNet is optimized better. Additionally, the joint losses is adopted as in Equation 4.39 in page 24 for optimization purposes.

4.3.3 Mining Dense Object Nets Representations in KeypointNet

The KeypointNet predicts semantic object keypoints, in turn equipping inert property for object generalization. DON computes dense generalized visual embeddings by exploiting the geometric prior of a sequence of registered RGBD frames to sample pixel correspondences between alternative views of the same object [28].

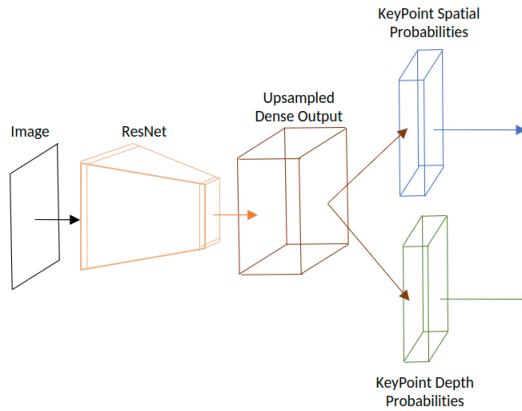
The KeypointNet shares the same architecture as DON ushering an idea that the KeypointNet might be encompassing generalized visual embeddings similarly computed by DON.

Intuitively, upsampled dense output features from the ResNet architecture in the KeypointNet might have the dense object representations as depicted in the Figure 4.4).

The KeypointNet is trained with the upsampled dense feature having output channel dimension of 256 and for benchmarking purposes, it is set to $D = 16$ such that it can be mutually benchmarked against DON computed representations.

The key advantage here is that the dense object representations can be directly extracted by upsampling the output of the last layer without the need of training DON saving time and resources. The same procedure is employed to store prioritized labels as in section 4.3.2 in page 24.

Figure 4.4: KeypointNet Architecture.



4.4 Generalizing “Caps”

The thesis moreover chooses to generalize caps. Blenderproc[43] is used to generate the cap dataset similar to ‘Suzanne’ dataset. 5 number of caps are chosen such that each of them have distinct shapes, designs and colors refer Table 4.2 from the Shapenet [35] library as it has rich object information including textures.

To make the training more robust such that networks are more object centric, the images are additionally augmented with random backgrounds and noisy backgrounds refer Figure in page in addition to color jitter and greyscaling augmentations. The mask information in the synthetic dataset is used to augment the backgrounds.

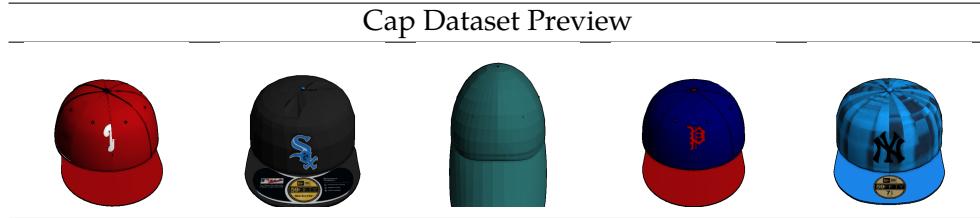
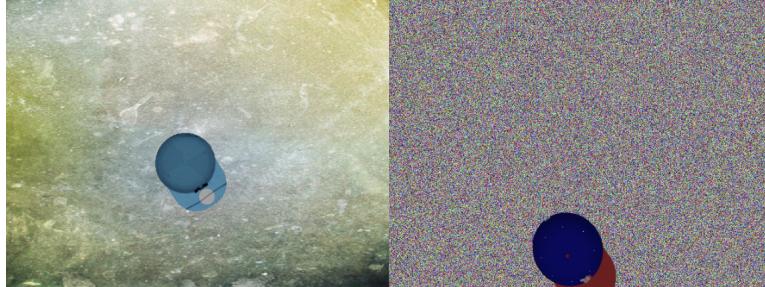


Table 4.2: Illustration of caps sampled from the Shapenet library.

Figure 4.5: The image in the right illustrates the noisy backgrounds and the image in left describes random background augmentation.



4.4.1 Semantic Correspondences Mapping Pipeline

By definition, KeypointNet regresses semantic keypoints on the objects. Meaning, the keypoint in one object is positioned semantically equivalent in another object which belongs to same class of the previous object.

Furthermore, this property produces correspondences in two image pairs of two different objects belonging to same class or semantically equivalent introducing a new method to train DON using semantic correspondences.

Loading semantic equivalent correspondences might affect the evalutaion of DON based on $PCK@k$ metric. To further evaluate DON, semantic correspondences will not be used instead, sequence of same objects are sampled with mutual correspondences using depth, camera intrinsics and extrinsics information to evaluate the metric.

4.4.2 Tweaking the Training Pipeline

Loading semantic correspondences to train DON calls for a revision in the existing training pipeline. At first, the KeypointNet is trained to compute semantic correspondences across objects belonging to the same class. The pretrained KeypointNet will be used to train DON. Finally, use the pretrained DON to train the KeypointNet to sample geometrically

consistent single class generalized labels for robot grasping.

“End-to-End” method of training the networks will not be used as the KeypointNet can only be trained using same object and not semantically equivalent object. Meanwhile, DON can be trained using the semantic correspondences.

4.4.3 Evaluating Single Class Generalized Labels in the Wild

The DON definitely produces single class generalized labels (dense descriptors) by definition and the KeypointNet being benchmarked for producing equivalent dense descriptors image, single class generalized labels can be accessed manually querying a pixel location in the dense descriptors image. Moreover, KeypointNet can be used to pick and store geometrically consistent labels based on keypoint locations.

“*In the Wild*” means that the sampled single class generalized labels will be evaluated against images which are not in the training pipeline. The images used for evaluation will be either fetched from the internet or actual images of caps lying around.

4.4.4 Robot Grasping Pipeline

To develop the robot grasping pipeline, Franka Emika 7-DOF robot manipulator is used. It is one of the robots available at sereact [50], Stuttgart, Germany equipped with hardware equipped to run artificial intelligence algorithms. The robot is equipped with the 2-jaw parallel gripper. Furthermore, Intel Realsense D435 camera is wrist mounted refer Figure 4.6.

The RGB image is streamed in to the networks to compute dense object descriptors from DON and 6d pose produced from keypoints from the KeypointNet. By using pre-computed single class generalized labels stored offline the Gaussian kernel is employed refer equation 4.20 in page 18 to produce grasping points for the robot. The depth information along with camera extrinsic and intrinsic information is used to project the spatial expectation of the queried label (descriptor) to camera frame coordinate for robot grasping.

4 Methodology

Figure 4.6: Depiction of robot setup for cap grasping with wrist mounted camera.



5 Results

5.1 Dense Object Nets

For benchmarking, 48GB VRAM GPU is used. As per the VRAM availability, the ResNet architecture with pixelwise distribution loss is evaluated with 2048 correspondences for an image pair with the descriptor dimension $D = 3$ and batch size of 2, and for the pixelwise nt-xent loss, 512 correspondences for an image pair with descriptor dimension $D = 16$ are evaluated. Each model is trained for 100 epochs with Adam optimizer [51] initialized with learning rate of 3×10^{-5} with weight regularization of 1×10^{-4} . Minimum validation loss is monitored to save best model. Each epoch took 45s. Training DON with the pixelwise distribution loss occupies less VRAM compared to the pixelwise nt-xent loss.

The $PCK@k$ metric Equation 4.14 is computed following the procedure described in [28]. The benchmark results are presented in Table 5.1, where the numbers in bold mark the optimal benchmark result (\uparrow signifies higher score is better). The optimal benchmark result is decided based minimum validation loss and computation costs.

The results have higher values compared to [28] as the model is evaluated for a single object scene rather than for a multi-object scene as described in [28]. The pixelwise distribution loss outperforms the pixelwise contrastive loss for a single object scene in accordance to the benchmark produced in [28].

Benchmarking DON for $AUC \pm \sigma$ for $PCK@K \uparrow$				
Loss Functions	D	ResNet-18	ResNet-34	ResNet-50
Pixelwise Distribution	3	0.9800	0.9800	0.9800
Pixelwise Distribution	16	0.9800	0.9800	0.9800
Pixelwise NT-Xent	3	0.9791 ± 0.0003	0.9796 ± 0.0003	0.9798 ± 0.0001
Pixelwise NT-Xent	16	0.9799 ± 0.0005	0.9800	0.9776 ± 0.0039

Table 5.1: DON evaluation against $AUC \pm \sigma$ for $PCK@K$ metric.

As for the descriptor dimension $D = 3$ the pixelwise distribution loss produces consistent results, as described in Table 5.1, it is evident that it produces optimal descriptors with respect to keypoint spatial expectation compared to the pixelwise nt-xent loss and is chosen for further implementation.

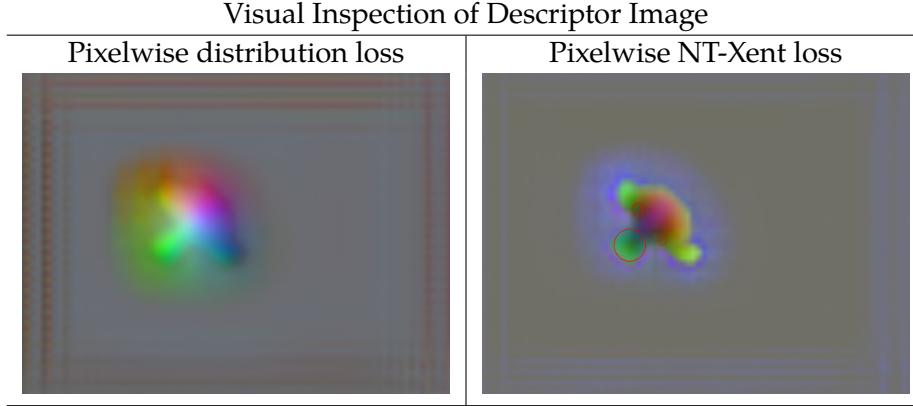


Table 5.2: Visual inspection for optimal placement of descriptors.

To compare both the loss functions visually, the dense descriptor image is extracted from the DON and color encoded by scaling the descriptor ($d \in \mathbb{R}^3$) to the pixel space of $p \in \mathbb{R}^3$, s.t. $p \in [0, 255]$. By definition, the dense descriptors are locally unique to each other. As described in Table 5.2, the descriptors encoded by the network trained on the pixelwise nt-xent loss yields the non-optimal descriptors (annotated in red circles in bottom image) compared to the network trained on pixelwise distribution loss which performs better.

5.1.1 Manual Method for Robot Grasp Generation

In the interactive application developed, the descriptors are selected manually. The manually selected descriptors pixels locations are projected to camera frame using depth and camera intrinsics to produce a 6D pose computed via PCA. The DON produces consistent descriptors against color jitter, occlusion, illumination and viewpoint refer Table 5.3. Furthermore, the 6D pose generated via the interactive application is not benchmarked as it is developed to test the consistency of the descriptors produced by DON.

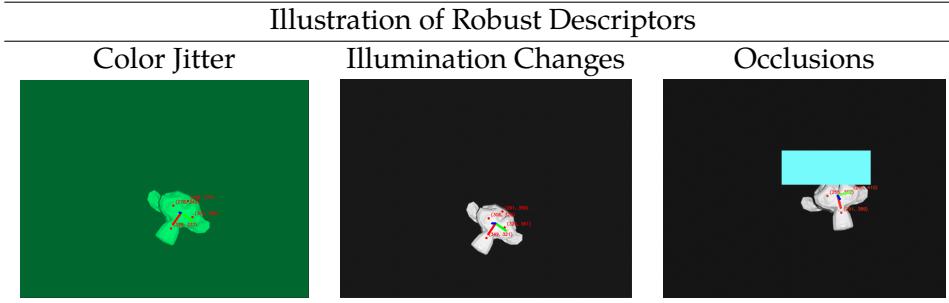


Table 5.3: Testing descriptors for robust 6D pose generation.

5.2 KeypointNet

The ResNet architecture is used to train KeypointNet task for 1000 epochs with batch size 16. To train the network, ADAM optimizer is used with the learning rate of 1×10^{-3} with no weight regularization. The optimal weights to compute to weighted sum loss in Equation 4.38 in page 23 is set to $W = [1.0, 0.2, 1.0, 1.0, 0.0001]^T$. The ResNet-18 and ResNet-34 architectures are trained to predict 8, 16, 32 number of geometrically consistent keypoints with $\delta = 10$ pixel spacing margin. The models are saved to monitor the minimum validation loss while training.

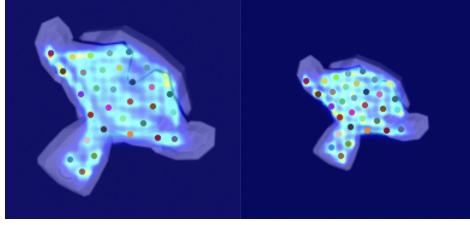
To benchmark the KeypointNet, the losses used to train the network itself is applied as it captures information like multiview consistency, relative pose, and whether a keypoint belongs to an object silhouette, which are the pivotal properties that a keypoint must carry. The Table 5.4 describes the benchmarking results sampled from 10 random image pairs. (\downarrow in the Table 5.4 represents that lower score is better.)

KeypointNet Benchmarking \downarrow					
Model	N	\mathcal{L}_{mc}	\mathcal{L}_{pose}	\mathcal{L}_{obj}	\mathcal{L}_{sep}
ResNet-18	8	0.1720 ± 0.0545	0.0826 ± 0.1123	0 ± 0	0 ± 0
ResNet-34	8	0.1115 ± 0.0244	0.1303 ± 0.1325	0 ± 0	0.0084 ± 0.0465
ResNet-18	16	0.2121 ± 0.0676	0.1427 ± 0.1332	0 ± 0	0.2745 ± 0.0576
ResNet-34	16	0.0830 ± 0.0891	0.0732 ± 0.0891	0.1813 ± 0.2142	0 ± 0
ResNet-18	32	1.0130 ± 0.2978	0.0990 ± 0.1276	1.3405 ± 0.3274	0.0088 ± 0.0234
ResNet-34	32	0.4206 ± 0.1906	0.0974 ± 0.0990	0.0928 ± 0.2440	0.0122 ± 0.0508
ResNet-50	32	0.1983 ± 0.6378	0.1312 ± 0.1262	1.3815 ± 0	0.2019 ± 0.0829

Table 5.4: Benchmarking KeypointNet with individual losses.

As per the benchmarking results in Table 5.4 in page 31, the ResNet-34 performs better overall. As per the ResNet-18 training history, the validation loss increases after some epochs and does not predict with consistent lower validation loss while, the ResNet-34 yields consistent decreasing in validation loss till the end of the training refer Figure 5.1. Additionally, ResNet-50 is benchmarked to predict 32 keypoints and found under performing. Hence, ResNet-34 architecture instead will be used for future implementation.

Figure 5.1: Output of trained ResNet-34 for KeypointNet task.



5.3 End-to-End Training

5.3.1 KeypointNet Informed Dense Object Nets

The ResNet-34 is used to implement KeypointNet tasks supplying 256 number of correspondences per image implying 512 correspondences for an image pair to train ResNet-18 based DON with margin $\delta = 3$ pixel apart. DON is trained with pixelwise distribution loss with the descriptor dimension $D = 3$. Adam optimizer is used to train the networks jointly with learning rate of 1×10^{-3} and no weight regularization. W for training the KeypointNet loss is set to the same value as used in KeypointNet benchmarking. The benchmarked results in Table 5.5 in page 32 are computed by extracting output features from the individual networks stacked to train in end-to-end training fashion.

KeypointNet Informed DON Benchmarking				
DON	$AUC \pm \sigma$ for PCK@Kmetric 0.9666 ± 0.0034			
KeypointNet	\mathcal{L}_{mc} 0.4367 ± 0.9020	\mathcal{L}_{pose} 0.0893 ± 0.1897	\mathcal{L}_{obj} 0.0019 ± 0.0020	\mathcal{L}_{sep} 0.0957 ± 0.0160

Table 5.5: Benchmarking KeypointNet informed DON for their individual performance.

The DON performance decreases compared to the benchmarking results in Table 5.1 in page 29. The keypoints computed from the KeypointNet are not consistent as one can see for \mathcal{L}_{mc} in Table 5.5. As the inconsistent keypoints are supplied to DON for training, the performance of DON decreases. Furthermore, the pixelwise distribution loss is not robust to inconsistent correspondence as claimed in [25].

5.3.2 Dense Object Nets Informed KeypointNet

The ResNet-18 is used to train DON of the descriptor dimension $D = 3$ with 256 correspondences per image implying 512 correspondences are sampled to train an image pair. The ResNet-34 is used to train KeypointNet predicting 8 keypoints on dense object representations with $\delta = 10$ pixels apart to pick labels. ADAM optimizer is used to train

5 Results

the networks jointly with learning rate of 1×10^{-3} and no weight regularization. W for training the KeypointNet loss is set to the same value as used in KeypointNet benchmarking. The benchmarked results in Table 5.6 in page 33 are computed by extracting output features from the individual networks stacked to train in end-to-end training fashion.

DON Informed KeypointNet Benchmarking				
DON	$AUC \pm \sigma$ for PCK@Kmetric			
	0.9756 ± 0.0005			
KeypointNet	\mathcal{L}_{mc}	\mathcal{L}_{pose}	\mathcal{L}_{obj}	\mathcal{L}_{sep}
	0.4323 ± 0.1671	0.1048 ± 0.0476	0.0029 ± 0.0050	0 ± 0

Table 5.6: Benchmarking results of DON informed KeypointNet with autopicked labels.

As per the benchmarking results in Table 5.6, the performance of DON decreases as the number of correspondences to train the DON network is limited to 256 for an image. This implies that all the loss functions to train the DON are sensitive to number of correspondences.

The Table 5.7 describes the single class generalized object labels autonomously picked by the KeypointNet from the dense descriptor image. The policy π assigns a random priority to store the labels for ‘Suzanne’ offline. The labels are queried from the descriptor space using 8 keypoints regressed by the KeypointNet with the separation margin $\delta = 10$ in pixel space.

Autosampled single class generalized object label	
π (Priority)	Label
1	$[-3.3778, -3.2668, 6.2417]^T$
2	$[-3.4086, -3.3175, 6.2570]^T$
3	$[-3.3793, -3.3144, 6.2799]^T$
4	$[-3.4003, -3.3314, 6.2663]^T$
5	$[-3.3072, -3.1737, 6.1999]^T$
6	$[-3.2575, -3.1253, 6.2076]^T$
7	$[-3.3644, -3.2777, 6.2714]^T$
8	$[-3.3931, -3.2843, 6.2413]^T$

Table 5.7: KeypointNet sampled class generalized labels for ‘Suzanne’.

5.3.3 Mining Dense Object Nets Representations in KeypointNet

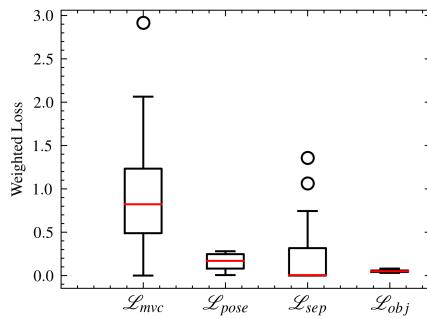
The KeypointNet is trained to regress 256 keypoints with $\delta = 2$ using ResNet-34 architecture with the same training setting to benchmark the KeypointNet. The upsampled dense output feature dimension is reduced to $D = 16$ for comparing with the DON whose

5 Results

descriptor dimension is set to $D = 16$.

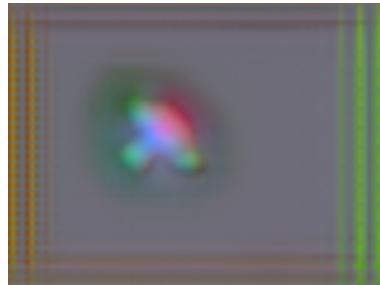
The dense representation put forth by the upsampled dense output feature layer is benchmarked against $PCK@$ as in Equation 4.14 in page 16, the same benchmarking process for DON. The upsampled dense output feature evaluates to $AUC \pm \sigma$ for $PCK@k, \forall k \in [1, 100] = 0.9759 \pm 0.0051$. This result inlines with the DON bechmarked against pixel-wise nt-xent loss with descriptor dimension $D = 16$. Futhermore, the loss for predicting the keypoints increased as illustrated in Figure ?? in page 34 as the upsampled dense output feature dimension is reduced from 256 to 16.

Figure 5.2: Errors in keypoints predictions from KeypointNet modified to benchmark against DON representations.



To represent the descriptor space image of dimension $D = 16$ in an image as in Figure 5.3, PCA is used to reduce the dimension to a higher dimensional sub-space of $D = 3$ (a higher dimensional sub-space is the lower dimensional representation of a higher dimension). PCA is only used for visualization purposes. The representation is not consistent as PCA preserves variance in the data while projecting it to a higher dimensional sub-space explaining the ears of ‘Suzanne’ with same color. No further attempts are carried out to represent dense descriptor images in an image as it does not aligned with the thesis objective.

Figure 5.3: Depiction of dense descriptors extracted from KeypointNet.



5.4 Generalizing “Caps”

5.4.1 Semantic Correspondence Mapping Pipeline

KeypointNet is trained to predict 100 keypoints with pixel spacing margin $\delta = 5$. Adam optimizer is used to train the network for 1000 epochs with a learning rate 1×10^{-3} and no weight regularization. The W is set similarly to the same value as in the KeypointNet benchmarking. Furthermore, Keypointnet is based on ResNet-34 architecture with up-sampled dense output dimension set to 256.

The KeypointNet is benchmarked for its loss function as detailed in Table 5.8 in page 35. The benchmarking yields higher errors compared to the results in Table 5.4 in page 31 as the number of correspondences is higher and semantically equivalent objects are trained. For the visual inspection purposes, 8 keypoints are sampled randomly as in the Table 5.9 in page 35. Most of the keypoints are semantically well placed, with a few keypoints regressed with multiview consistent errors.

Semantic Correspondence Mapping Pipeline Benchmarking			
\mathcal{L}_{mc}	\mathcal{L}_{pose}	\mathcal{L}_{obj}	\mathcal{L}_{sep}
5.1342 ± 2.8517	0.1456 ± 0.0628	0.2135 ± 0.0771	2.0256 ± 1.2637

Table 5.8: Benchmarking: KeypointNet used for mapping semantic correspondences.

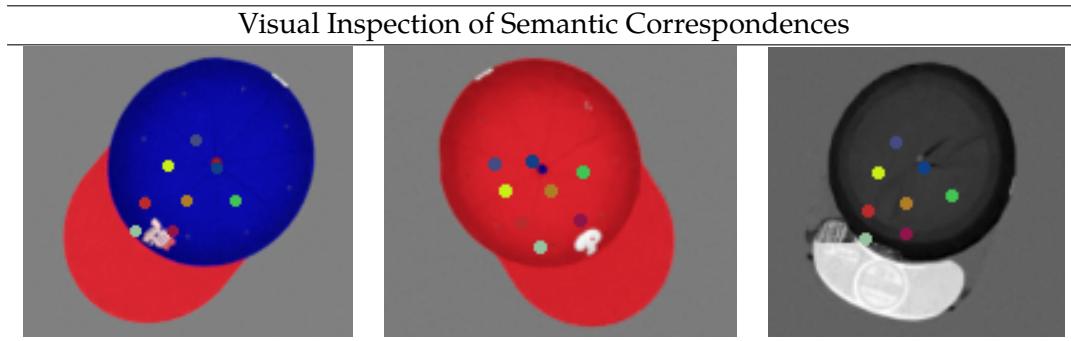


Table 5.9: Illustration of semantic correspondences. The semantic correspondences are color coded.

Furthermore, the semantic correspondence pipeline is examined for the root cause of the high errors. The semantic correspondence pipeline did not converge for a cap in the synthetic dataset. As illustrated in Figure 5.4 in page 36, the keypoints do not meet the properties of geometrically consistent keypoints further, explaining the high errors in Table 5.8 in page 35. The object is removed from the dataset, and the semantic

5 Results

correspondence pipeline is trained again to resolve predictions of keypoints with errors. The errors decrease after another training as can be seen in Table 5.10 in page 36.

Figure 5.4: Non-Convergence in Semantic Correspondence Pipeline.



Revised Semantic Correspondence Mapping Pipeline Benchmarking			
\mathcal{L}_{mc}	\mathcal{L}_{pose}	\mathcal{L}_{obj}	\mathcal{L}_{sep}
2.1210 ± 1.5937	0.0754 ± 0.1840	0.0780 ± 0.0079	0.1374 ± 1.8752

Table 5.10: Revised bechmarking for semantic correspondences mapping pipeline.

5.4.2 Evaluating Single Class Generalized Labels in the Wild

Two images of caps were collected using a smartphone with a camera to benchmark single-label generalizing ability in the wild. Furthermore, from now on, benchmarking method initially used to quantify the neural network tasks will not be used.



Table 5.11: Depiction of two caps found in the facility.

For evaluating DON trained on the synthetic dataset of caps, DON is trained with ResNet-34 architecture with pixelwise correspondence loss with the dense descriptor dimension $D = 3$ as it is previously found to be performing optimally as can be seen in Table 5.1 in

5 Results

page 29 and computationally economical. Additionally, DON is trained on the semantic correspondences supplied from the KeypointNet, and ADAM optimizer is used with a learning rate of 3×10^{-5} with weight regularization set to 1×10^{-4} for 100 epochs with batch size 2.

The trained DON is tested on the cap images. We demonstrate its optimal descriptor placement in Table 5.11 in page 36. Judging from Figure 5.5, the descriptors are not placed semantically well. The heatmaps in the image describe a descriptor's spatial probability in two images. The heatmaps are generated from the interactive application developed to generate robot grasps manually with the temperature parameter set to $t = -0.3$ in Gaussian kernel search space equation 4.20 in page 18.

Figure 5.5: Testing optimal descriptor placement for the caps in the wild.



Furthermore, the DON is benchmarked for the synthetic dataset. As illustrated in the Figure 5.6 in page 38, the DON performs optimally. The DON places spatially optimal descriptors semantically across the objects in the same class.

The synthetic dataset is perfect without any faults compared to the real-world data. Despite using background randomizations, noisy backgrounds, colour jitters and greyscaling augmentations, the DON performs poorly on the real-world data as the descriptors are not placed spatially optimally in the semantically same objects.

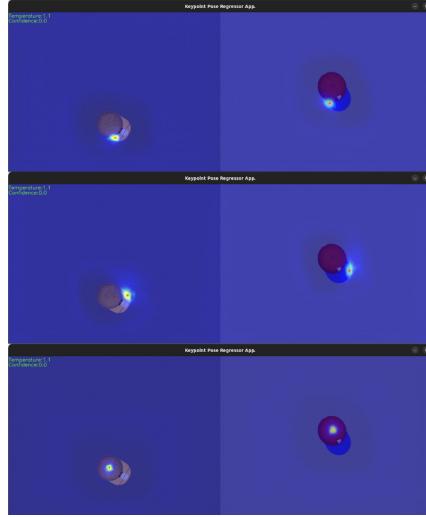
5.4.3 “Not losing hope on the real world”

As per the results, the neural networks perform optimally locally in synthetic data and not real data, further posing an issue for robot grasping.

The neural network implementation scheme has three parts. i.e., semantic correspondence mapping pipeline, DON and KeypointNet to pick auto labels. Synthetic correspondences are adopted from [28] to train DON, replacing the method of geometrically computing correspondences from depth images.

5 Results

Figure 5.6: DON performing optimally on the synthetic dataset



To compute synthetic correspondences, the spatial grid \mathcal{G} as in Equation 4.37 in page 22 is stacked with the RGB image such that the image dimension is as follows:

$$I \rightarrow \mathbb{R}^{H \times W \times 5} \quad (5.1)$$

Furthermore, the image is affinely augmented with previously used augmentations from the vision library provided by [52]. Correspondences are the intersection of the spatial grid in the augmented image and the anchor image as illustrated in the Figure 5.7 in page 38. These synthetic correspondences are further consumed to train DON.

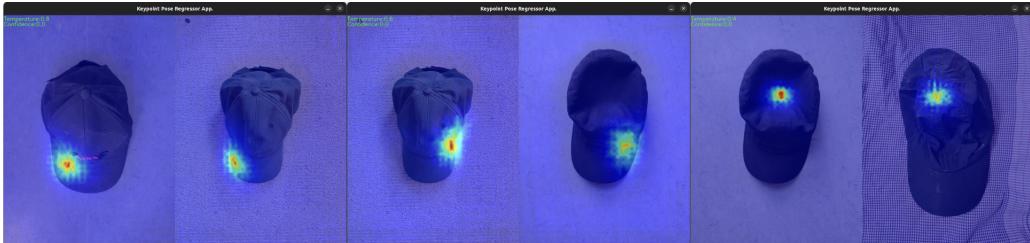
Figure 5.7: Illustrations of synthetic correspondences computed from the image augmentation. The pixel annotated in blue in the left image corresponds to the same pixel annotated in green in the right image.



5 Results

The synthetic correspondences generalize caps in the real world pretty well, as depicted in the Figure 5.8 in page 5.8.

Figure 5.8: Visual inspection of DON trained on synthetic correspondences.



5.4.4 Robot Grasping Pipeline

The KeypointNet is trained on the DON representations to regress 3 geometrically consistent keypoints on dense descriptor images. These keypoint locations are the location of the dense descriptor labels. Furthermore, these labels are stored offline in the .json format as rendered in the Listing 5.1 in page 39 for querying the descriptors in the real-time application.

```

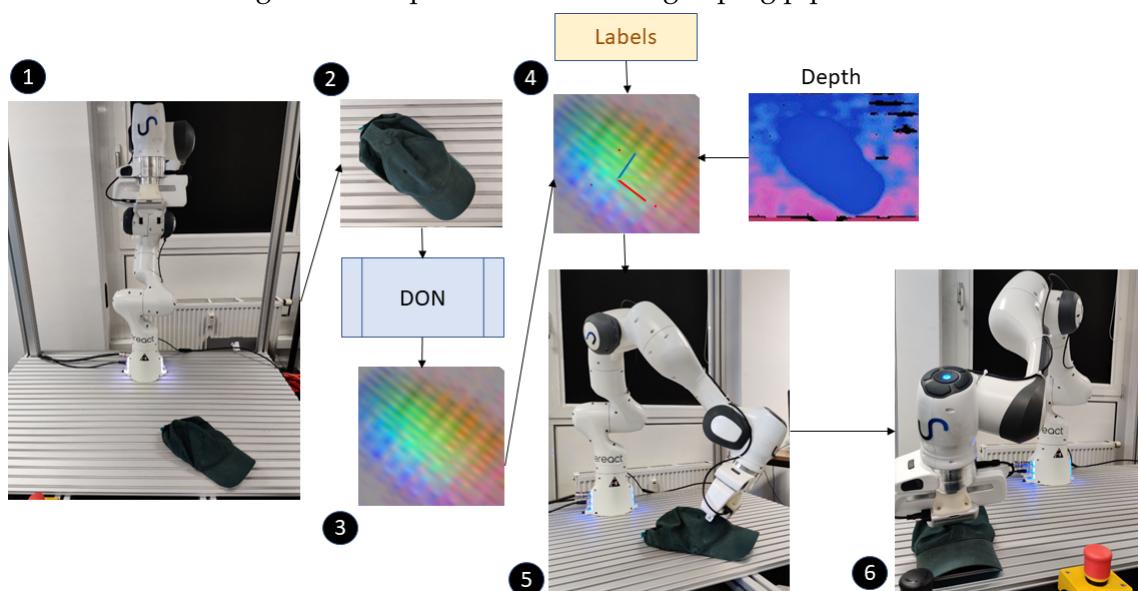
1  ["cap": {
2      "labels": [
3          [3.1030, -3.4000, 2.9119],
4          [5.2890, -2.5363, -0.7921],
5          [0.0596, -3.8049, 2.1469]
6      ]
7      "priority": [
8          1,
9          2,
10         3
11     ]
12 }
13 ]
```

Listing 5.1: Preview of json file entry of offline stored descriptors.

The robot grasping pipeline is illustrated in the Figure 5.9. The robot samples an RGB image highlighted as 1 in the image. The sampled RGB image is processed by the DON marked as 2 to create a dense descriptor image highlighted as 3 in the figure. Using labels stored offline along with the depth information, 6D pose of the objects is computed highlighted as 4. The robot grasps the object by adjusting the gripper position according to the 6D pose for robust grasping and placing as highlighted in 5 and 6 in the Figure 5.9.

5 Results

Figure 5.9: Depiction of the robot grasping pipeline.



6 Conclusion

The thesis set off with the aim of creating artificial intelligence for robots like Chappie and C-3PO. These robots can perform generalized tasks rather than task-oriented activities, which is the present state of today's artificial intelligence. Moreover, the thesis focused on developing generalized representations of an object in an image for robot manipulation solving the present industrial robotics problem of teaching robots to pick every object equipped with 2D or 3D vision systems with the need for explicit programming for each object.

Set on to implement artificial intelligence with generalizing capabilities, DON is implemented. The DON is introduced to the robotics community by Florence *et al.* [7] with generalization capabilities of objects in an image. The DON generates dense descriptor image from an RGB image capable of generalizing an object for robot manipulation. Furthermore, the thesis benchmarked various loss functions, different ResNet neural networks, and as per the results, the "Pixelwise Distribution Loss" with ResNet-34 performs the best to generalize objects belonging to the same class (i.e., semantically equivalent) and computationally economical compared to "Pixelwise NT-Xent Loss". Single class generalized labels manually extracted from the pixels of the dense descriptor image provided by DON are applied to various applications like generating robust object 6D poses of objects in cases of illumination changes, colour changes, occlusion and different viewpoints and other application being robot grasping.

To eliminate the need of manually selecting labels from the dense descriptor image put forth by the DON, KeypointNet is implemented. The KeypointNet, as described, predicts a set of oriented geometrically consistent keypoints on an object such that it preserves the object's 6D pose.

The KeypointNet comes with generalizing properties as it predicts geometrically consistent keypoints across objects belonging to the same class. The generalizing property of KeypointNet is exploited to create a semantic correspondence pipeline to train the DON, replacing the need for depth information which is often noisy with today's technology of depth cameras available. Additionally, the KeypointNet is trained on manifold-based loss making it faster to converge and the need of an orientation network is eliminated. Furthermore, the manifold loss implemented to train the KeypointNet limits the object rotation to a range of $[0, \pi]$. The ResNet-34 architecture performs optimally for the predicting geometrically consistent keypoints. The KeypointNet is sensitive to occlusion

6 Conclusion

and fails to predict keypoints on occluded areas of objects encountered in the training phase.

As DON is robust against occlusion, KeypointNet could benefit from it. KeypointNet trained on DON representations, regresses a keypoint that occupies a pixel in the dense descriptor image later where these pixel locations are queried as single class generalized labels eliminating any need for human intervention.

The upsampled dense output put forth by the KeypointNet demonstrated in Figure 4.4 in page 25 represents dense object representations as an alternative to DON to create single-class generalized labels. This ushers an idea that there are other networks capable of computing dense descriptors image similarly to DON.

Initially, the networks are trained on synthetic data. The synthetic data is too perfect to be generalized on real-world objects. The KeypointNet and DON trained on the synthetic dataset did not perform well in the wild. The end-to-end training of neural networks showed reduced performance when trained with the synthetic dataset.

Synthetic correspondences introduced in [28] are used to overcome the limitations of training the networks posed by the synthetic data. The synthetic correspondences are independent of camera intrinsic and extrinsic information, including depth information to compute correspondences. The DON is trained on images of caps captured from a smartphone with a camera with synthetic correspondence and performed well, generalizing most of the caps in the wild.

The capabilities of DON and KeypointNet are demonstrated and applied to generalized autonomous robot task of cap pick-and-place operation.

7 Future Scope

As the Keypointnet is limited by the manifold losses for training objects with rotation in range $[0, \pi]$, a newer formulation will be engineered and applied. Furthermore, KeypointNet and DON will be expanded to accommodate objects belonging to different classes.

Bibliography

- [1] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, *et al.*, “A general reinforcement learning algorithm that masters chess, shogi, and go through self-play”, *Science*, vol. 362, no. 6419, pp. 1140–1144, 2018.
- [2] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, *et al.*, “Mastering the game of go with deep neural networks and tree search”, *nature*, vol. 529, no. 7587, pp. 484–489, 2016.
- [3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks”, *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.
- [4] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation”, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.
- [5] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask r-cnn”, in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.
- [6] R. A. Güler, N. Neverova, and I. Kokkinos, “Densepose: Dense human pose estimation in the wild”, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7297–7306.
- [7] P. R. Florence, L. Manuelli, and R. Tedrake, “Dense object nets: Learning dense visual object descriptors by and for robotic manipulation”, *arXiv preprint arXiv:1806.08756*, 2018.
- [8] P. Sundaresan, J. Grannen, B. Thananjeyan, A. Balakrishna, M. Laskey, K. Stone, J. E. Gonzalez, and K. Goldberg, “Learning rope manipulation policies using dense object descriptors trained on synthetic depth data”, *CoRR*, vol. abs/2003.01835, 2020. arXiv: 2003 . 01835.
- [9] C.-Y. Chai, K.-F. Hsu, and S.-L. Tsao, “Multi-step pick-and-place tasks using object-centric dense correspondences”, in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 4004–4011.
- [10] P. Florence, L. Manuelli, and R. Tedrake, “Self-supervised correspondence in visuomotor policy learning”, *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 492–499, 2019.

Bibliography

- [11] A. Ganapathi, P. Sundaresan, B. Thananjeyan, *et al.*, "Learning dense visual correspondences in simulation to smooth and fold real fabrics", in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 11 515–11 522.
- [12] A. Kupcsik, M. Spies, A. Klein, M. Todescato, N. Waniek, P. Schillinger, and M. Bürger, "Supervised training of dense object nets using optimal descriptors for industrial robotic applications", *arXiv preprint arXiv:2102.08096*, 2021.
- [13] B. Pal, S. Khaiyum, and Y. S. Kumaraswamy, "3d point cloud generation from 2d depth camera images using successive triangulation", in *2017 International Conference on Innovative Mechanisms for Industry Applications (ICIMIA)*, 2017, pp. 129–133.
- [14] Y. Song and H. Yan, "Image segmentation techniques overview", in *2017 Asia Modelling Symposium (AMS)*, 2017, pp. 103–107.
- [15] S. Noor, M. Waqas, M. I. Saleem, and H. N. Minhas, "Automatic object tracking and segmentation using unsupervised siammask", *IEEE Access*, vol. 9, pp. 106 550–106 559, 2021.
- [16] A. Maćkiewicz and W. Ratajczak, "Principal components analysis (pca)", *Computers & Geosciences*, vol. 19, no. 3, pp. 303–342, 1993.
- [17] V. Klema and A. Laub, "The singular value decomposition: Its computation and some applications", *IEEE Transactions on Automatic Control*, vol. 25, no. 2, pp. 164–176, 1980.
- [18] S. Ruder, "An overview of multi-task learning in deep neural networks", *CoRR*, vol. abs/1706.05098, 2017. arXiv: 1706 . 05098.
- [19] S. Suwanjanakorn, N. Snavely, J. J. Tompson, and M. Norouzi, "Discovery of latent 3d keypoints via end-to-end geometric reasoning", *Advances in neural information processing systems*, vol. 31, 2018.
- [20] T. Schmidt, R. Newcombe, and D. Fox, "Self-supervised visual descriptor learning for dense correspondence", *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 420–427, 2016.
- [21] D. G. Lowe, "Object recognition from local scale-invariant features", in *Proceedings of the seventh IEEE international conference on computer vision*, Ieee, vol. 2, 1999, pp. 1150–1157.
- [22] X. Zhao, R. Vemulapalli, P. A. Mansfield, B. Gong, B. Green, L. Shapira, and Y. Wu, "Contrastive learning for label efficient semantic segmentation", in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 10 623–10 633.
- [23] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation", *Neural computation*, vol. 15, no. 6, pp. 1373–1396, 2003.
- [24] D. Hadjivelichkov and D. Kanoulas, "Fully self-supervised class awareness in dense object descriptors", in *5th Annual Conference on Robot Learning*, 2021.

Bibliography

- [25] P. R. Florence, "Dense visual learning for robot manipulation", Ph.D. dissertation, Massachusetts Institute of Technology, 2020.
- [26] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations", in *International conference on machine learning*, PMLR, 2020, pp. 1597–1607.
- [27] J. Zbontar, L. Jing, I. Misra, Y. LeCun, and S. Deny, "Barlow twins: Self-supervised learning via redundancy reduction", in *International Conference on Machine Learning*, PMLR, 2021, pp. 12310–12320.
- [28] D. B. Adrian, A. G. Kupcsik, M. Spies, and H. Neumann, "Efficient and robust training of dense object nets for multi-object robot manipulation", in *2022 International Conference on Robotics and Automation (ICRA)*, IEEE, 2022, pp. 1562–1568.
- [29] L. Manuelli, W. Gao, P. R. Florence, and R. Tedrake, "Kpam: Keypoint affordances for category-level robotic manipulation", *CoRR*, vol. abs/1903.06684, 2019. arXiv: 1903.06684.
- [30] M. Vecerik, J.-B. Regli, O. Sushkov, D. Barker, R. Pevceviciute, T. Rothörl, C. Schuster, R. Hadsell, L. Agapito, and J. Scholz, "S3k: Self-supervised semantic keypoints for robotic manipulation via multi-view consistency", *arXiv preprint arXiv:2009.14711*, 2020.
- [31] L. Yen-Chen, A. Zeng, S. Song, P. Isola, and T.-Y. Lin, "Learning to see before learning to act: Visual pre-training for manipulation", in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2020, pp. 7286–7293.
- [32] P. Sermanet, C. Lynch, J. Hsu, and S. Levine, "Time-contrastive networks: Self-supervised learning from multi-view observation", in *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, IEEE, 2017, pp. 486–487.
- [33] Y. Li, G. Wang, X. Ji, Y. Xiang, and D. Fox, "Deepim: Deep iterative matching for 6d pose estimation", in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 683–698.
- [34] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes", *arXiv preprint arXiv:1711.00199*, 2017.
- [35] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, et al., "Shapenet: An information-rich 3d model repository", *arXiv preprint arXiv:1512.03012*, 2015.
- [36] L. Yen-Chen, P. Florence, J. T. Barron, T.-Y. Lin, A. Rodriguez, and P. Isola, *Nerf-supervision: Learning dense object descriptors from neural radiance fields*, 2022.
- [37] A. Simeonov, Y. Du, A. Tagliasacchi, J. B. Tenenbaum, A. Rodriguez, P. Agrawal, and V. Sitzmann, *Neural descriptor fields: Se(3)-equivariant object representations for manipulation*, 2021.

Bibliography

- [38] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition”, pp. 770–778, 2016.
- [39] S. I. Nikolenko, *Synthetic data for deep learning*. Springer, 2021, vol. 174.
- [40] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, “A simple framework for contrastive learning of visual representations”, in *International conference on machine learning*, PMLR, 2020, pp. 1597–1607.
- [41] A. v. d. Oord, Y. Li, and O. Vinyals, “Representation learning with contrastive predictive coding”, *arXiv preprint arXiv:1807.03748*, 2018.
- [42] C.-Y. Chai, K.-F. Hsu, and S.-L. Tsao, “Multi-step pick-and-place tasks using object-centric dense correspondences”, in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2019, pp. 4004–4011.
- [43] M. Denninger, M. Sundermeyer, D. Winkelbauer, Y. Zidan, D. Olefir, M. Elbadrawy, A. Lodhi, and H. Katam, “Blenderproc”, *arXiv preprint arXiv:1911.01911*, 2019.
- [44] S. Kelly, “Basic introduction to pygame”, in *Python, PyGame and Raspberry Pi Game Development*, Springer, 2016, pp. 59–65.
- [45] R. Bhatia, *Matrix analysis*. Springer Science & Business Media, 2013, vol. 169.
- [46] D. Q. Huynh, “Metrics for 3d rotations: Comparison and analysis”, *Journal of Mathematical Imaging and Vision*, vol. 35, no. 2, pp. 155–164, 2009.
- [47] J. M. Lee, *Introduction to Riemannian manifolds*. Springer, 2018, vol. 176.
- [48] W. Kabsch, “A solution for the best rotation to relate two sets of vectors”, *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography*, vol. 32, no. 5, pp. 922–923, 1976.
- [49] C. B. Choy, J. Gwak, S. Savarese, and M. Chandraker, “Universal correspondence network”, *Advances in neural information processing systems*, vol. 29, 2016.
- [50] “ai robots for production. today.” sereact.
- [51] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization”, *arXiv preprint arXiv:1412.6980*, 2014.
- [52] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al., “Pytorch: An imperative style, high-performance deep learning library”, *Advances in neural information processing systems*, vol. 32, 2019.

List of Acronyms

PCA	Principal Component Analysis
SVD	Singular Value Decomposition
CNN	Convolutional Neural Network
DON	Dense Object Nets
SOTA	State of the Art
SIFT	Scale-Invariant Feature Transform
NeRF	Neural Radiance Fields
RGB	3-Channel Image
ResNet	Residual Network
s.t.	such that
DNN	Deep Neural Network
DL	Deep Learning
RGB-D	4-Dimensional Image, i.e., RGB with Depth

List of Figures

4.1	DON architecture.	12
4.2	2D spatial expectation of keypoint in two images.	15
4.3	Variance loss function in work.	22
4.4	KeypointNet Architecture.	25
4.5	The image in the right illustrates the noisy backgrounds and the image in left describes random background augmentation.	26
4.6	Depiction of robot setup for cap grasping with wrist mounted camera.	28
5.1	Output of trained ResNet-34 for KeypointNet task.	32
5.2	Errors in keypoints predictions from KeypointNet modified to benchmark against DON representations.	34
5.3	Depiction of dense descriptors extracted from KeypointNet.	34
5.4	Non-Convergence in Semantic Correspondence Pipeline.	36
5.5	Testing optimal descriptor placement for the caps in the wild.	37
5.6	DON performing optimally on the synthetic dataset	38
5.7	Illustrations of synthetic correspondences computed from the image augmentation. The pixel annotated in blue in the left image corresponds to the same pixel annotated in green in the right image.	38
5.8	Visual inspection of DON trained on synthetic correspondences.	39
5.9	Depiction of the robot grasping pipeline.	40

List of Tables

2.1	Comparison of influential SOTA.	10
4.1	Depiction of dataset quintennials features of dataset	17
4.2	Illustration of caps sampled from the Shapenet library.	26
5.1	DON evaluation against $AUC \pm \sigma$ for $PCK@K$ metric.	29
5.2	Visual inspection for optimal placement of descriptors.	30
5.3	Testing descriptors for robost 6D pose generation.	30
5.4	Benchmarking KeypointNet with individual losses.	31
5.5	Benchmarking KeypointNet informed DON for their individual performance.	32
5.6	Benchmarking results of DON informed KeypointNet with autopicked labels.	33
5.7	KeypointNet sampled class generalized labels for ‘Suzanne’.	33
5.8	Benchmarking: KeypointNet used for mapping semantic correspondences.	35
5.9	Illustration of semantic correspondences. The semantic correspondences are color coded.	35
5.10	Revised bechmarking for semantic correspondences mapping pipeline. .	36
5.11	Depiction of two caps found in the facility.	36

Listings

5.1 Preview of json file entry of offline stored descriptors. 39

List of Symbols

Symbol	Description
\mathbf{D}	Dense descriptor dimension
\mathbb{N}^+	Set of positive integers
\in	Belongs to or in
\mathcal{M}	Riemannian manifold
$SO(n)$	Special orthogonal group of dimension n
$SE(3)$	Special euclidean group of dimension 3
\mathbb{R}^n	Real number of dimension n
$\langle \cdot, \cdot \rangle$	Scalar product of two vectors
$\ \cdot \ $	L^2 Norm
$\ \cdot \ _F$	Frobenius Norm