
Training Dense Object Nets: A Novel Approach

Anonymous Author(s)

Affiliation
Address
email

Abstract

1 Our work proposes a novel framework that addresses the computational limitations
2 associated with training Dense Object Nets (DON) while achieving robust and
3 dense visual object descriptors. DON's descriptors are known for their robustness
4 to viewpoint and configuration changes, but training these requires image pairs
5 with computationally expensive correspondence mapping. This limitation hampers
6 dimensionality and robustness, thereby restricting object generalization. To over-
7 come this, we introduce data generation procedure using synthetic augmentation
8 and a novel deep learning architecture that produces denser visual descriptors
9 with reduced computational demands. Notably, our framework eliminates the
10 need for image pair correspondence mapping and showcases its application in a
11 robotic grasping pipeline. Experimental results demonstrate that our approach
12 yields descriptors as robust as those generated by DON.

13

1 Introduction

14 Creating a general-purpose robotic system capable of performing practical tasks, such as those
15 portrayed in movies like Chappie [1] or C-3PO [2], is a significant challenge in robotics. While
16 advancements have been made in related domains, achieving this goal remains an ongoing endeavour.
17 Recent artificial intelligence (AI) breakthroughs, particularly in deep learning, have demonstrated
18 remarkable capabilities. For instance, AlphaGo [3], an AI system trained entirely on self-play,
19 defeated the world's best human Go player at the time. Subsequent algorithms, such as those
20 developed by Silver et al. [4], have mastered complex games like chess, Go, World of Warcraft [5],
21 and Shogi, surpassing human expertise. These achievements underscore the importance of visual
22 data in deep learning, as these algorithms learn directly from visual inputs like gameplay recordings
23 or online video streams. Additionally, the introduction of AlexNet [6] in 2012 has revolutionized
24 computer vision, leading to significant progress in tasks like semantic segmentation [7], object
25 identification and recognition [8], and human pose estimation [9].
26 Significant strides have also been made in robotics, including developing self-driving cars and
27 humanoid robots capable of complex tasks using cameras and vision sensors. Integration of AI
28 models, such as ChatGPT [10] and PaLM [11], has further enhanced the capabilities of robots.
29 ChatGPT, developed by OpenAI, improves human-robot interactions by generating coherent and
30 contextually relevant responses, enabling robots to engage in meaningful conversations and provide
31 informative answers. Palm-E, developed by Stanford University, combines computer vision and deep
32 reinforcement learning, enabling robots to perceive and manipulate objects in their environment.
33 These AI models hold significant potential to enhance the dexterity and adaptability of robots, opening
34 up new possibilities across various industries.
35 However, developing and deploying these sophisticated AI models, particularly Large Language
36 Models (LLMs) like ChatGPT and Palm-E, present their own challenges. LLMs' sheer size and
37 complexity demand vast computational resources, raising concerns about excessive energy consump-
38 tion and environmental impact. Researchers such as Bender et al. [12] and Strubell et al. [13] have

39 raised alarms about the substantial carbon emissions associated with training and operating LLMs at
40 scale, as well as the exacerbation of the digital divide due to the resource-intensive nature of LLMs.
41 Responsible resource allocation and addressing the ethical implications of prioritizing computational
42 power for LLMs are crucial for balancing technological progress and sustainability.

43 Currently, in robotics, typical industrial robots perform repetitive operations based on pre-programmed
44 instructions, finding the ideal object representation for grasping and manipulation tasks still needs
45 to be answered. Existing representations may be unable to understand an object's geometrical
46 and structural information, rendering them unsuitable for complex tasks In recent work, Florence
47 et al. [14] introduced a novel visual object representation termed "dense visual object descriptors" to
48 the robotics community. This representation, generated by the Dense Object Nets (DON) framework,
49 converts each pixel in an image ($I[u, v] \in \mathbb{R}^3$) into a higher-dimensional embedding ($I_D[u, v] \in \mathbb{R}^D$)
50 such that $D \in \mathbb{N}^+$, using image-pair correspondences as input. These dense visual object descriptors
51 provide a generalized representation of objects to a certain extent.

52 The DON framework has shown promise in various domains, including rope manipulation [15], block
53 manipulation [16], robot control [17], fabric manipulation [18], and robot grasp pose estimation [19,
54 20]. Adrian et al. [20] demonstrated that DON can be trained on synthetic data and still generalize
55 well to real-world objects. Furthermore, they highlighted the importance of the dimensionality of the
56 embedding in determining the quality of the descriptors produced by the DON framework.

57 In this paper, we address the challenge posed by the computationally intensive nature of DON and
58 propose a new framework for training DON in a computationally efficient manner. Furthermore, we
59 introduce a novel synthetic data generation pipeline that generates a complete dataset from one image
60 and mask pair. Additionally, the synthetic data generation pipeline does not rely on the noisy depth
61 information produced by today's consumer-grade depth cameras. We also demonstrate one of the
62 applications of our framework as a robotic grasping pipeline. Our approach aims to contribute to
63 developing a sustainable and efficient, and economical solution for industrial robotics applications.

64 2 Related Work

65 Florence et al. [14] introduced the Pixelwise Contrastive loss function to train DON, which involves
66 sampling pixels in an image-pair and computing the Contrastive loss between the pixels in the first
67 image and those in the second image. This optimization procedure aims to improve the descriptor
68 based on a similarity metric. However, the Pixelwise Contrastive loss function is computationally
69 expensive and requires numerous matching and non-matching image-pair correspondences to work
70 optimally. When optimizing DON using a large number (N) of image-pair correspondences, the
71 computational resources consumed by the optimization procedure increase significantly due to the
72 exponential growth of pixelwise descriptor similarity comparisons (2^N).

73 In their work, Florence [21] discovered that the Pixelwise Contrastive loss function used to train
74 DON might yield poor performance if a computed correspondence is spatially inconsistent. They
75 also highlighted that the precision of contrastive-trained models could be sensitive to the relative
76 weighting between positive and negative sampled pixels. To address these limitations, Florence
77 proposed a new continuous sampling-based loss function called the Pixelwise Distribution loss. This
78 novel loss function leverages smooth and continuous pixel space sampling instead of the discrete pixel
79 space sampling method employed by the Pixelwise Contrastive loss. The Pixelwise Distribution loss
80 eliminates the need for non-matching correspondences, leading to significant savings in computation
81 resources.

82 On a different note, Kupcsik et al. [19] utilized Laplacian Eigenmaps [22] to embed a 3D object model
83 into an optimally generated embedding space, serving as the target for training DON in a supervised
84 fashion. However, this methodology does not reduce the computational resource consumption
85 required to train DON. In contrast, Hadjivelichkov and Kanoulas [23] employed offline unsupervised
86 clustering based on confidence in object similarities to generate hard and soft correspondence labels.
87 These labels were then used as matching and non-matching correspondences to train DON effectively.

88 Building upon the concept of SIMCLR-inspired frameworks [24, 25], Adrian et al. [20] introduced
89 a similar architecture and another novel loss function called the Pixelwise NTXent Loss. This loss
90 function robustly trains DON by leveraging synthetic correspondences computed from image augmen-
91 tations and non-matching image correspondences. Notably, Adrian et al.'s experiments demonstrated

92 that the novel loss function is invariant to batch size variations, unlike the Pixelwise Contrastive Loss.
93 Furthermore, it is worth noting that most of the discussed optimization methodologies heavily rely on
94 correspondences to train DON effectively.

95 Moving on to the aspect of image-pair correspondences and dataset engineering, the DON training
96 strategy proposed in [14, 21] relies on depth information to compute correspondences between image
97 pairs using camera intrinsics and pose information [26]. However, when utilizing consumer-grade
98 depth cameras to capture depth information, the resulting depth data can be noisy, particularly when
99 dealing with tiny, reflecting objects common in industrial environments. Noisy depth information
100 hampers the computation of consistent spatial correspondences in an image pair. To overcome
101 this challenge, Kupcsik et al. [19] associated 3D models of objects with image views, effectively
102 training DON without relying on depth information. Their approach proved efficient for smaller,
103 texture-less, and reflective objects. Additionally, Kupcsik et al. compared different training strategies
104 for producing 6D grasps on industrial objects and demonstrated that a unique supervised training
105 approach enhances pick-and-place resilience in industry-relevant tasks.

106 In contrast, Yen-Chen et al.[27] employed NeRF[28], a method that reconstructs a 3D scene from
107 a sequence of images captured by a smartphone camera. They extracted correspondences from the
108 synthetically reconstructed scene to train DON. Remarkably, Adrian et al.’s experiments indicated
109 that DON trained on synthetic data generalizes well to real-world objects. Furthermore, they adopted
110 the $PCK@k$ metric, commonly used in [29, 30], to evaluate and benchmark DON’s performance in
111 cluttered scenes that were previously not extensively studied.

112 In our work, we do not use any loss functions as proposed in [14, 21, 19, 20, 23, 27] to train DON.
113 However, we adopt the network architecture from DON [14] as our architecture’s backbone and train
114 on the task of the KeypointNet[31, 32] with few network modifications. Moreover, we evaluate the
115 descriptor’s robustness produced by our framework on the $PCK@k$ metric as in comparision to
116 benchmarks in [20] as it is the only benchmark available for DON. Furthermore, we compare the
117 computational resource consumption.

118 3 Methodology

119 In this section, we outline the methodologies employed. Our approach encompasses synthetic dataset
120 engineering, a novel framework, loss function modifications and a comprehensive grasping pipeline.
121 Firstly, we focus on synthetic dataset engineering accomodating spatial, colour and background
122 augmentation. The color and background augmentations help the framework to predict object-oriented
123 descriptors. Secondly, we present a novel framework designed to reduce computational resource
124 consumption with loss function modifications to optimize performance. Lastly, we introduce a
125 comprehensive robot grasping pipeline, integrating all methodologies into a unified system.

126 3.1 Dataset Engineering

127 We have chosen the cap object for creating a synthetic dataset as the cap mesh models are readily
128 available in the Shapenet library [33] as it contains rich object information, including textures.
129 Furthermore, we choose five cap models from the Shapenet library and use Blenderproc [34] to
130 generate the synthetic dataset. We save one RGB image, mask, and depth for each cap model from the
131 synthetic scene. Additionally, we employ synthetic augmentations as proposed in [20] to synthetically
132 spatial augment the cap’s position and rotation in an image, including background randomization using
133 Torchvision [35] library. An augmented image-pair is sampled randomly to generate camera poses for
134 different viewpoints. Additionally, image-pair correspondences are computed¹ as illustrated in the
135 Figure 1. Using depth information, we project the computed correspondences to the camera frame and
136 compute the relative transformation between two camera-frame coordinates of the correspondences
137 using Kabsch’s transformation [36]. Moreover, mask and depth images are not used during inference.

¹GitHub Link:
<https://github.com/KanishkNavale/Mapping-Synthetic-Correspondences-in-an-Image-Pair>



Figure 1: Depiction of image synthetic spatial augmentation and correspondences mapping in an image-pair. The colored encoded dots in the figure represents correspondences in an image-pair.

138 3.2 Framework & Mining Strategy

139 As a backbone, we employ ResNet-34 architecture [37]. We preserve the last convolution layer and
 140 remove the pooling and linear layers. The backbone downsamples the RGB image $I_{RGB} \in \mathbb{R}^{H \times W \times 3}$
 141 to dense features $I_d \in \mathbb{R}^{h \times w \times D}$ such that $h \ll H, w \ll W$ and $D \in \mathbb{N}^+$. We upsample the dense
 142 features from the identity layer (being identical to the last convolution layer in the backbone) as
 143 illustrated in the Figure 2 in page 4 as follows:

$$f_U : I \in \mathbb{R}^{h \times w \times D} \rightarrow I_D \in \mathbb{R}^{H \times W \times D}. \quad (1)$$

144 The upsampled dense features are extracted and treated as dense visual local descriptors produced
 145 from the DON. In otherwords we extract or mine the representations from the backbone. Similarly as
 146 in [31], we stack spatial-probability regressing layer and depth regressing layer on top of the identity
 147 layer to predict $N \in \mathbb{N}^+$ number of keypoint's spatial-probability as follows:

$$f_S : I_d \in \mathbb{R}^{h \times w \times D} \rightarrow I_s^N \in \mathbb{R}^{h \times w \times N}, \quad (2)$$

148 and depth as follows:

$$f_D : I_d \in \mathbb{R}^{h \times w \times D} \rightarrow I_{\hat{d}} \in \mathbb{R}^{h \times w \times N}. \quad (3)$$

149 We incorporate continuous sampling method f_E from [21, 31] to convert the upsampled predicted
 150 spatial-probability and depth of a keypoint to spatial-depth expectation as follows:

$$f_E \circ g_E : [I_s, I_{\hat{d}}] \rightarrow [u, v, d]^T \in \mathbb{R}^3, \text{ where } g_E : I \in \mathbb{R}^{h \times w \times N} \rightarrow I \in \mathbb{R}^{H \times W \times N}. \quad (4)$$

151 Furthermore, we train the framework in a twin architecture fashion as proposed in [24, 25, 14, 21, 19,
 152 20, 23, 27] on the modified KeypointNet task.

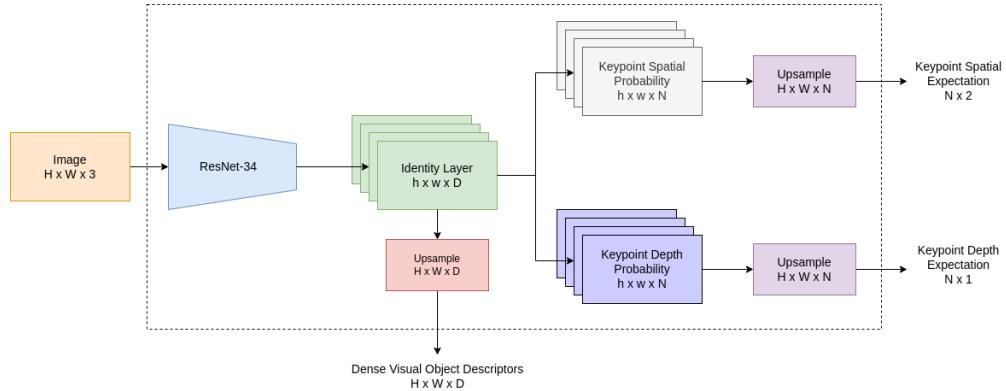


Figure 2: Illustration of the novel framework designed to compute and seamlessly extract dense visual object descriptors efficiently. During inference, we extract dense visual object descriptors directly from the network and ignore predicted spatial-depth expectations of the keypoints.

153 **3.3 Loss Functions**

154 For training, we directly adopt silhouette consistency loss (\mathcal{L}_{obj}), variance loss (\mathcal{L}_{var}) and separation
 155 loss (\mathcal{L}_{sep}) functions from [31] to train the network on the keypoint prediction task. However, we
 156 modify the multi-view consistent loss and relative pose estimation loss. In the case of multi-view
 157 consistency loss we project the predicted spatial-depth expectation using camera intrinsics as follows:
 158

$$X_{cam} \in \mathbb{R}^{3 \times 1} = \mathcal{I}_{cam}^{-1} [u, v, 1.0]^T \times d, \text{ where } \mathcal{I}_{cam} \in \mathbb{R}^{3 \times 3} \text{ and } u, v, d \in \mathbb{R}^+. \quad (5)$$

159 Furthermore, we project the camera coordinates of the keypoints from one camera viewpoint to
 160 another camera viewpoint using relative transformation supplied from the synthetic augmentation
 161 procedure as follows:

$$\mathcal{L}_{mvc} \in \mathbb{R} = \mathcal{H}(\hat{X}_{cam}^B, \mathcal{T}_{A \rightarrow B} \hat{X}_{cam}^A), \text{ where } \hat{X}_{cam} = [X_{cam}, 1.0]^T \in \mathbb{R}^{4 \times 1}, \quad (6)$$

162 In Equation 6, $\mathcal{T}_{A \rightarrow B} \in SE(3)$ is a Special Euclidean Group [38] which is relative transformation
 163 from camera-frame A to camera-frame B . We use Huber loss \mathcal{H} as it produces smoother gradients
 164 for framework optimization. Furthermore, we do not discard the relative transformation information
 165 to calculate the relative pose loss as suggested in [31]. Moreover, being influenced from [32] we
 166 modified the relative pose loss as follows:

$$\mathcal{L}_{pose} = \|\log(\mathcal{T}_{truth}^\dagger \mathcal{T}_{pred})\|, \text{ where } \log : SE(3) \rightarrow \mathfrak{se}(3) \text{ and } \mathcal{T}^\dagger = \begin{bmatrix} R^T & -R^T t \\ 0^T & 1 \end{bmatrix}. \quad (7)$$

167 **3.4 Robot Grasping Pipeline**

168 To use the proposed framework as a robot grasping pipeline, we extract dense visual object descriptors
 169 from the network and store one single descriptor of objects in a database manually for now. During
 170 inference, we extract dense visual object descriptors from the network and query the descriptor from
 171 the database to find the closest match as follows:

$$\mathbb{E}[u^*, v^*]_d = \underset{u, v}{\operatorname{argmin}} \exp - \left(\frac{\|I_D[u, v] - d\|}{\exp(t)} \right)^2, \text{ where } \|I_D[u, v] - d\| \in \mathbb{R}^{H \times W}. \quad (8)$$

172 Where $t \in \mathbb{R}$ controls the kernel width influencing the search space to compute the optimal spatial
 173 expectation $\mathbb{E}[u^*, v^*]_d$ of the query descriptor $d \in \mathbb{R}^D$ in the descriptor image $I_D \in \mathbb{R}^{H \times W \times D}$.
 174 The computed spatial expectation is projected to the robot frame using camera intrinsics and poses
 175 to perform a pinch grasp. Furthermore, the Franka Emika 7-DOF robot manipulator with two jaw
 176 gripper and wrist-mounted Intel Realsense D435 camera is used as a testing setup as illustrated in
 177 Figure 3.



Figure 3: Illustration of the robot grasping pipeline setup. In the image, the robot is highlighted in red, the caps in green, and the camera in blue.

178 **4 Experiments & Results**

179 In this section, we outline the benchmarking results employed from the methodologies. We benchmark
 180 the DON framework with Pixelwise NT-Xent loss as in [20] and our framework with our revision of
 181 the loss function. We benchmark the descriptor’s robustness with the $AUC \pm \sigma$ for $PCK@k, \forall k \in$
 182 $[1, 100]$ metric. Furthermore, we benchmark the computational resource consumption of the DON
 183 and our frameworks. We also demonstrate the application of our framework as a robot-grasping
 184 pipeline in two methodologies, one of which our framework demonstrates its capabilities to produce
 185 object-specific 6D poses for robot grasping.

186 **4.1 Dense Object Nets**

187 We implemented training and benchmarking using “PyTorch-Lightning”[39] and “PyTorch”[40]
 188 libraries. Furthermore, we employ ADAM[41] optimizer to optimize the model for 2500 epochs
 189 with a learning rate of $\alpha = 3 \times 10^{-4}, \beta_1 = 0.9$ and $\beta_2 = 0.999$ with weight decay $\eta = 10^{-4}$ to
 190 benchmark the DON with Pixelwise NT-Xent loss as in [20] with a fixed batch size of 1 and 128
 191 image-pair correspondences. As per the benchmarking results in Table 1, the descriptor’s robustness
 192 increases as the descriptor’s dimension gets longer.

Table 1: Benchmark of DON framework for GPU consumption and $AUC \pm \sigma$ for $PCK@k, \forall k \in [1, 100]$ metric.

DON benchmark				
Descriptor Size (D)	3	8	16	32
AUC for $PCK@k$	0.922 ± 0.006	0.933 ± 0.011	0.948 ± 0.012	0.953 ± 0.008
VRAM Usage (GB)	9.377	13.717	20.479	30.067

193 The $AUC \pm \sigma$ for $PCK@k, \forall k \in [1, 100]$ is computed with 256 image-pair correspondences and
 194 the metrics mean and std. deviation is calculated from benchmarking 3 DON models trained for a
 195 single descriptor dimension. We could not train the descriptor dimension of 64 and 128 due to the
 196 limited VRAM. Furthermore, to inspect the results of trained DON, an interface is built using the
 197 PyGame library [42] to visualize the results of the trained DON. The mouse pointer in the image
 198 space is mapped to the pixel, and the descriptor at that pixel is queried in another image-descriptor
 199 space. We further use the spatial probability of the descriptor to visualize the queried descriptor in the
 200 image space using Equation 8 in page 5. We identify if there are any multi-modal spatial activations
 201 in the descriptor spaces and none, as shown in Figure 4.

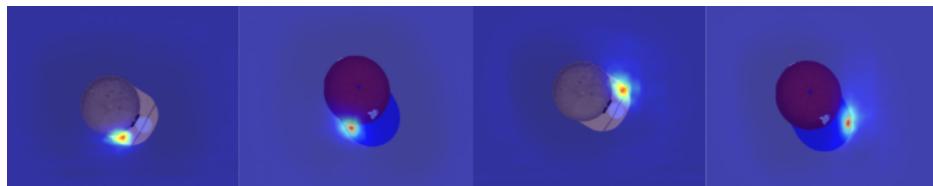


Figure 4: Depiction of the spatial probability heatmaps of the descriptor in the image space. We set the temperature in the Equation 8 to 1.1 and render the spatial probability heatmaps in the interface. The first and second image from the left and the right highlights the semantically equivalent descriptors in the image space.

202 **4.2 Our Framework**

203 To train our framework, we employ an ADAM optimizer to optimize the model for 2500 epochs
 204 with a learning rate of $\alpha = 1 \times 10^{-3}, \beta_1 = 0.9$ and $\beta_2 = 0.999$ with no weight decay. We further
 205 use a fixed batch size of 1 and the StepLR scheduler with a step size 2500 and a gamma of 0.9 to
 206 train the model with all the loss weights to 1.0 except variance loss weight to 1×10^{-3} . We trained
 207 the three models with 128 keypoints with a margin of 2 pixels for each descriptor dimension. We
 208 specifically chose 128 keypoints as it aligns with the notion that DON is benchmarked with 128

209 image-pair correspondences. As per the Table 2 in page 7, the descriptor’s robustness increases as the
 210 descriptor’s dimension gets longer, which is in line with the benchmarking results of DON.

Table 2: Benchmark of our framework for GPU consumption and $AUC \pm \sigma$ for $PCK@k, \forall k \in [1, 100]$ metric.

Our framework with 128 keypoints		
Descriptor Size (D)	$AUC \pm \sigma$ for $PCK@k$	VRAM Usage (GB)
3	0.914 ± 0.009	4.763
8	0.928 ± 0.015	4.785
16	0.945 ± 0.010	4.832
32	0.950 ± 0.009	4.872
64	0.953 ± 0.006	4.913
128	0.957 ± 0.012	5.409
256	0.959 ± 0.008	6.551
512	0.962 ± 0.011	7.915

211 To check the impact of descriptors’ robustness compared to the number of keypoints, we trained our
 212 framework with 16 keypoints. Furthermore, we trained three additional models for each descriptor
 213 dimension 64, 128, 256, and 512. As per the table 3 compared to the results in Table 2, the descriptor’s
 214 robustness decreased when the framework predicted 16 keypoints. Moreover, this reflects that number
 215 of keypoints in our framework and the number of image-pair correspondences in DON are directly
 216 proportional to the robustness of the descriptors.

Table 3: Benchmark of our framework with 16 keypoints for GPU consumption and $AUC \pm \sigma$ for
 $PCK@k, \forall k \in [1, 100]$ metric.

Our framework with 16 keypoints		
Descriptor Size (D)	$AUC \pm \sigma$ for $PCK@k$	VRAM Usage (GB)
64	0.948 ± 0.009	3.799
128	0.952 ± 0.010	4.191
256	0.955 ± 0.013	5.241
512	0.957 ± 0.006	7.341

217 4.3 Robot Grasping Pipeline

218 For the robot grasping pipeline, we trained our framework with actual caps. As the synthetic data
 219 generation only needs mask and depth information, we could create a mask in no time. Additionally,
 220 while training the framework, we do not need the actual real-world depth information as it computes
 221 its own. We later extracted the dense visual local descriptors from the framework. We visually
 222 inspected for any inconsistencies in the descriptor space, as shown in Figure 5, and found it consistent.
 223 Furthermore, we did not use the models trained on the synthetic dataset, as the representations were
 224 inconsistent with the real caps.

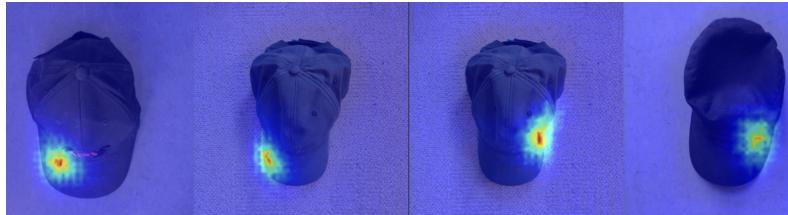


Figure 5: Visual inspection of the dense visual descriptors space of the real caps.

225 For robot grasping, a descriptor is picked from the descriptor space and queried in real-time such that
 226 robot can pinch-grasp the object. We could successfully grasp the caps with the robot, as shown in
 227 Figure 6 in page 8.

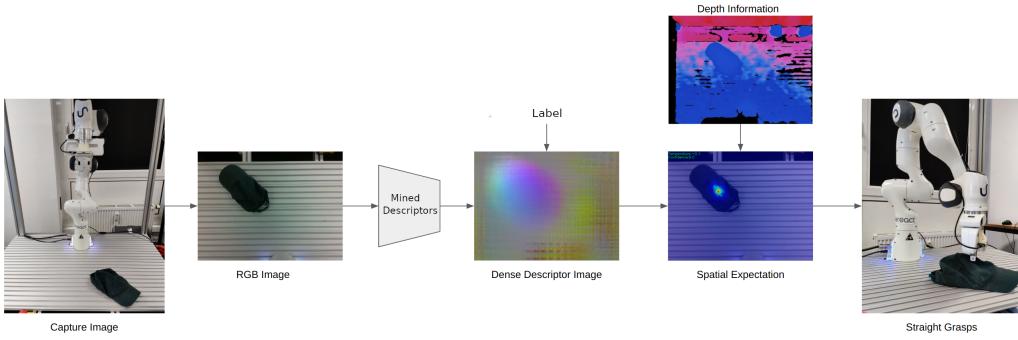


Figure 6: Depiction of the straight robot grasping pipeline.

228 As our framework inertly regresses keypoints on the object, we could use it as an alternative approach
 229 to grasp the caps by computing the pose generated by the keypoints considering the actual depth
 230 information instead of network-regressed depth information. We extract the spatial probability of
 231 each keypoint from the framework and deactivate spatial probabilities where the depth information is
 232 missing, as the depth image from the camera is noisy. Furthermore, the spatial expectations of the
 233 keypoints are projected to the camera frame to calculate a 6D pose in the camera frame. The 6D pose
 234 is transformed in the robot frame to perform an aligned grasp, as shown in Figure 7.

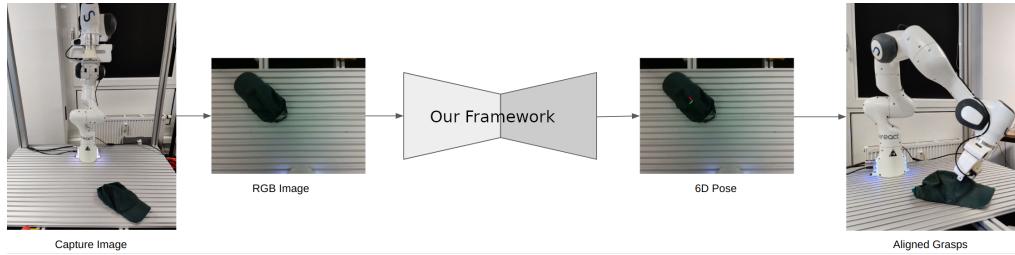


Figure 7: Illustration of the aligned robot grasping pipeline.

235 5 Conclusion

236 This paper introduces a novel framework for mining dense visual object descriptors without explicitly
 237 training DON. We have successfully eliminated the requirement for image-pair correspondence
 238 mapping in training DON by employing synthetic augmentation data generation and a novel deep-
 239 learning architecture. Our benchmarking results showcase the effectiveness of our framework in
 240 generating robust and denser visual local descriptors. However, it needs to outperform the original
 241 DON framework in robustness. Moreover, a notable advantage of our proposed framework is its
 242 significantly reduced computational resource consumption, amounting to a remarkable 86.67%
 243 decrease compared to the originally proposed framework. It is important to note that our current
 244 framework is limited to single object-dense visual descriptors. Nevertheless, we have plans to
 245 extend our methodology to encompass the production of multi-object dense visual descriptors in
 246 cluttered scenes. By doing so, we aim to enhance the versatility and applicability of our framework in
 247 real-world scenarios. To demonstrate the practicality of our framework, we have integrated it into a
 248 robot-grasping pipeline using two distinct methodologies. Remarkably, our framework can generate
 249 object-specific 6D poses, enhancing robot grasping performance. This successful application further
 250 highlights the potential utility of our framework in real-world robotic systems.

251 6 Future Work

252 The framework will be extended to produce multi-object dense visual descriptors in cluttered scenes.
 253 Furthermore, we will start incorporating ROI layers in the framework and adding additional object
 254 classification loss functions.

255 **References**

- 256 [1] N. Blomkamp, H. Zimmer, and S. Kinberg. *Chappie*. Sony Pictures Home Entertainment, 2015.
- 257 [2] G. Lucas and S. Ulstein. *Star wars*. 20th Century-Fox, 1977.
- 258 [3] 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”. In: *Science* 362.6419 (2018), pp. 1140–1144.
- 261 [4] 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”. In: *nature* 529.7587 (2016), pp. 484–489.
- 264 [5] B. Entertainment. *World of warcraft*. Insight Editions, Division Of Palac, 2013.
- 265 [6] A. Krizhevsky, I. Sutskever, and G. E. Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Communications of the ACM* 60.6 (2017), pp. 84–90.
- 267 [7] 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.
- 269 [8] 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.
- 271 [9] 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.
- 273 [10] OpenAI. *GPT-4 Technical Report*. 2023. arXiv: 2303.08774 [cs.CL].
- 274 [11] A. Chowdhery et al. “PaLM: Scaling Language Modeling with Pathways”. In: *arxiv:2204.02311* (2022).
- 275 [12] E. M. Bender, T. Gebru, A. McMillan-Major, and S. Shmitchell. “On the Dangers of Stochastic Parrots: Can Language Models Be Too Big?” In: *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*. 2021, pp. 610–623.
- 278 [13] E. Strubell, A. Ganesh, and A. McCallum. “Energy and policy considerations for deep learning in NLP”. In: *arXiv preprint arXiv:1906.02243* (2019).
- 280 [14] P. R. Florence, L. Manuelli, and R. Tedrake. “Dense object nets: Learning dense visual object descriptors by and for robotic manipulation”. In: *arXiv preprint arXiv:1806.08756* (2018).
- 282 [15] 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”. In: *CoRR* abs/2003.01835 (2020). arXiv: 2003.01835.
- 285 [16] 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. DOI: 10.1109/IROS40897.2019.8968294.
- 288 [17] P. Florence, L. Manuelli, and R. Tedrake. “Self-supervised correspondence in visuomotor policy learning”. In: *IEEE Robotics and Automation Letters* 5.2 (2019), pp. 492–499.
- 290 [18] A. Ganapathi 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. 11515–11522. DOI: 10.1109/ICRA48506.2021.9561980.
- 293 [19] 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”. In: *arXiv preprint arXiv:2102.08096* (2021).
- 296 [20] 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.
- 299 [21] P. R. Florence. “Dense visual learning for robot manipulation”. PhD thesis. Massachusetts Institute of Technology, 2020.
- 301 [22] M. Belkin and P. Niyogi. “Laplacian eigenmaps for dimensionality reduction and data representation”. In: *Neural computation* 15.6 (2003), pp. 1373–1396.
- 303 [23] D. Hadjivelichkov and D. Kanoulas. “Fully Self-Supervised Class Awareness in Dense Object Descriptors”. In: *5th Annual Conference on Robot Learning*. 2021.
- 305 [24] 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.
- 307 [25] 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.
- 310 [26] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.
- 312 [27] 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. DOI: 10.48550/ARXIV.2203.01913.

- 314 [28] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. “Nerf: Representing
 315 scenes as neural radiance fields for view synthesis”. In: *Communications of the ACM* 65.1 (2021), pp. 99–
 316 106.
- 317 [29] C.-Y. Chai, K.-F. Hsu, and S.-L. Tsao. “Multi-step pick-and-place tasks using object-centric dense
 318 correspondences”. In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.
 319 IEEE. 2019, pp. 4004–4011.
- 320 [30] M. E. Fathy, Q.-H. Tran, M. Z. Zia, P. Vernaza, and M. Chandraker. “Hierarchical metric learning and
 321 matching for 2d and 3d geometric correspondences”. In: *Proceedings of the european conference on*
 322 *computer vision (ECCV)*. 2018, pp. 803–819.
- 323 [31] S. Suwajanakorn, N. Snavely, J. J. Tompson, and M. Norouzi. “Discovery of latent 3d keypoints via
 324 end-to-end geometric reasoning”. In: *Advances in neural information processing systems* 31 (2018).
- 325 [32] W. Zhao, S. Zhang, Z. Guan, W. Zhao, J. Peng, and J. Fan. “Learning deep network for detecting 3d
 326 object keypoints and 6d poses”. In: *Proceedings of the IEEE/CVF Conference on computer vision and*
 327 *pattern recognition*. 2020, pp. 14134–14142.
- 328 [33] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song,
 329 H. Su, et al. “Shapenet: An information-rich 3d model repository”. In: *arXiv preprint arXiv:1512.03012*
 330 (2015).
- 331 [34] M. Denninger, M. Sundermeyer, D. Winkelbauer, Y. Zidan, D. Olefir, M. Elbadrawy, A. Lodhi, and
 332 H. Katam. “Blenderproc”. In: *arXiv preprint arXiv:1911.01911* (2019).
- 333 [35] S. Marcel and Y. Rodriguez. “Torchvision the machine-vision package of torch”. In: *Proceedings of the*
 334 *18th ACM international conference on Multimedia*. 2010, pp. 1485–1488.
- 335 [36] W. Kabsch. “A solution for the best rotation to relate two sets of vectors”. In: *Acta Crystallographica*
 336 *Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography* 32.5 (1976), pp. 922–
 337 923.
- 338 [37] K. He, X. Zhang, S. Ren, and J. Sun. “Deep residual learning for image recognition”. In: (2016), pp. 770–
 339 778.
- 340 [38] W. P. Thurston. “Three-Dimensional Geometry and Topology, Volume 1”. In: *Three-Dimensional Geome-*
 341 *try and Topology, Volume 1*. Princeton university press, 2014.
- 342 [39] W. A. Falcon. “Pytorch lightning”. In: *GitHub* 3 (2019).
- 343 [40] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein,
 344 L. Antiga, et al. “Pytorch: An imperative style, high-performance deep learning library”. In: *Advances in*
 345 *neural information processing systems* 32 (2019).
- 346 [41] D. P. Kingma and J. Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint*
 347 *arXiv:1412.6980* (2014).
- 348 [42] S. Kelly. “Basic introduction to pygame”. In: *Python, PyGame and Raspberry Pi Game Development*.
 349 Springer, 2016, pp. 59–65.