Dense Object Nets: Learning Dense Visual Object Descriptors By and For Robotic Manipulation

Peter R. Florence*, Lucas Manuelli*, Russ Tedrake
CSAIL, Massachusetts Institute of Technology
{peteflo,manuelli,russt}@csail.mit.edu
*These authors contributed equally to this work.

Abstract: What is the right object representation for manipulation? We would like robots to visually perceive scenes and learn an understanding of the objects in them that (i) is task-agnostic and can be used as a building block for a variety of manipulation tasks, (ii) is generally applicable to both rigid and non-rigid objects, (iii) takes advantage of the strong priors provided by 3D vision, and (iv) is entirely learned from self-supervision. This is hard to achieve with previous methods: much recent work in grasping does not extend to grasping specific objects or other tasks, whereas task-specific learning may require many trials to generalize well across object configurations or other tasks. In this paper we present Dense Object Nets, which build on recent developments in self-supervised dense descriptor learning, as a consistent object representation for visual understanding and manipulation. We demonstrate they can be trained quickly (approximately 20 minutes) for a wide variety of previously unseen and potentially non-rigid objects. We additionally present novel contributions to enable multi-object descriptor learning, and show that by modifying our training procedure, we can either acquire descriptors which generalize across classes of objects, or descriptors that are distinct for each object instance. Finally, we demonstrate the novel application of learned dense descriptors to robotic manipulation. We demonstrate grasping of specific points on an object across potentially deformed object configurations, and demonstrate using class general descriptors to transfer specific grasps across objects in a class.

Keywords: Visual Descriptor Learning, Self-Supervision, Robot Manipulation

1 Introduction

What is the right object representation for manipulation? While task-specific reinforcement learning can achieve impressively dexterous skills for a given specific task [1], it is unclear which is the best route to efficiently achieving many different tasks. Other recent work [2, 3] can provide very general grasping functionality but does not address specificity. Achieving specificity, the ability to accomplish specific tasks with specific objects, may require solving the data association problem. At a coarse level the task of identifying individual objects to manipulate can be solved by instance segmentation, as demonstrated in the Amazon Robotics Challenge (ARC) [4, 5] or [6]. Whole-object-level segmentation, however, does not provide any information on the rich structure of the objects themselves, and hence may not be an appropriate representation for solving more complex tasks. While not previously applied to the robotic manipulation domain, recent work has demonstrated advances in learning dense pixel level data association [7, 8], including self-supervision from raw RGBD data [8], which inspired our present work.

In this paper, we propose and demonstrate using dense visual description as a representation for robotic manipulation. We demonstrate the first autonomous system that can entirely self-supervise to learn consistent dense visual representations of objects, and the first system we know of that is capable of performing the manipulation demonstrations we provide. Specifically, with no human supervision during training, our system can grasp specific locations on deformable objects, grasp semantically corresponding locations on instances in a class, and grasp specific locations on specific instances in clutter. Towards this goal, we also provide practical contributions to dense visual descriptor learning with general computer

Code, data, and video available: github.com/RobotLocomotion/pytorch-dense-correspondence

2nd Conference on Robot Learning (CoRL 2018), Zurich, Switzerland.

vision applications outside of robotic manipulation. We call our visual representations Dense Object Nets, which are deep neural networks trained to provide dense (pixelwise) description of objects.

Contributions. We believe our largest contribution is that we introduce dense descriptors as a representation useful for robotic manipulation. We've also shown that self-supervised dense visual descriptor learning can be applied to a wide variety of potentially non-rigid objects and classes (47 objects so far, including 3 distinct classes), can be learned quickly (approximately 20 minutes), and can enable new manipulation tasks. In example tasks we grasp specific points on objects across potentially deformed configurations, do so with object instance-specificity in clutter, or transfer specific grasps across objects in a class. We also contribute novel techniques to enable multi-object distinct dense descriptors, and show that by modifying the loss function and sampling procedure, we can either acquire descriptors which generalize across classes of objects, or descriptors that are distinct for each object instance. Finally, we contribute general training techniques for dense descriptors which we found to be critical to achieving good performance in practice.

Paper Organization. In Section 2 we describe related work. As preliminary in Section 3.1 we describe the general technique for self-supervising dense visual descriptor learning, which is from [8] but reproduced here for clarity. We then describe additional techniques we've developed for object-centric visual descriptors in Section 3.2, and Section 3.3 describes techniques for distinct multi-object descriptors. Section 4 describes our experimental setup for our autonomous system, and Section 5 describes our results: our learned visual descriptors for a wide variety of objects (Section 5.1) multi-object descriptors and selective class generalization (Sections 5.2 and 5.3), and robotic manipulation demonstrations (Section 5.4).

2 Related Work

We review three main areas of related work: learned descriptors, self-supervised visual learning for robots, and robot learning for specific tasks. The task of correspondence estimation from multiple views of the same scene is fundamental in computer vision, whereas dense semantic correspondence across *different* scenes was popularized by [9]. Recent advances have been made by introducing a pixel-wise variant of contrastive loss [10] combined with deep convolutional networks, as in Choy et al. [7] and Schmidt et al. [8]. For cross-instance semantic correspondence, [7] relies on human annotations, while [8] learns these unsupervised, as we do here. Other work [11] uses image warping to learn descriptors, and most require manually annotated labels [13, 14, 15]. Zeng et al. [12] also uses dense 3D reconstruction to provide automated labeling, but for descriptors of 3D volume patches. Some of these works [8, 11, 15], like ours, learn descriptors for specific object instances or classes, while others [12] learn descriptors for establishing correspondence of arbitrary data. None of these prior works in dense visual learning involve robots.

In the area of self-supervised visual robot learning, while some recent work has sought to understand 'how will the world change given the robot's action?" [16, 17] in this work we instead ask "what is the current visual state of the robot's world?". We address this question with a dense description that is consistent across viewpoints, object configurations and (if desired) object classes. At the coarse level of semantic segmentation several works from the Amazon Robotics Challenge used robots to automate the data collection and annotation process through image-level background subtraction [18, 5, 4]. In contrast this work uses 3D reconstruction-based change detection and dense pixelwise correspondences, which provides a much richer supervisory signal for use during training.

In the area of robot learning for a specific task there have been impressive works on end-to-end reinforcement learning [1, 19]. In these papers the goal is to learn a specific task, encoded with a reward function, whereas we learn a general task agnostic visual representation. There have also been several works focusing on grasping from RGB or depth images [3, 18, 2, 20]. These papers focus on successfully grasping any item out of a pile, and are effectively looking for graspable features. They have no consistent object representation or specific location on that object, and thus the robotic manipulation tasks we demonstrate in Section 5.4, e.g. grasping specific points on an object across potentially deformed object configurations, are out of scope for these works.

3 Methodology

3.1 Preliminary: Self-Supervised Pixelwise Contrastive Loss

We use self-supervised pixelwise contrastive loss, as developed in [7, 8]. This learns a dense visual descriptor mapping which maps a full-resolution RGB image, $\mathbb{R}^{W \times H \times 3}$ to a dense descriptor space, $\mathbb{R}^{W \times H \times D}$,

where for each pixel we have a D-dimensional descriptor vector. Training is performed in a Siamese fashion, where a pair of RGB images, I_a and I_b are sampled from one RGBD video, and many pixel matches and non-matches are generated from the pair of images. A pixel $u_a \in \mathbb{R}^2$ from image I_a is a match with pixel u_b from image I_b if they correspond to the same vertex of the dense 3D reconstruction (Figure 1 (c-f)). The dense descriptor mapping is trained via pixelwise contrastive loss. The loss function aims to minimize the distance between descriptors corresponding to a match, while descriptors corresponding to a non-match should be at least a distance M apart, where M is a margin parameter. The dense descriptor mapping $f(\cdot)$ is used to map an image $I \in \mathbb{R}^{W \times H \times 3}$ to descriptor space $f(I) \in \mathbb{R}^{W \times H \times D}$. Given a pixel u we use $f(I)(u)$ to denote the descriptor corresponding to pixel u in image I. We simply round the real-valued pixel $u \in \mathbb{R}^2$ to the closest discrete pixel value $u \in \mathbb{N}^2$, but any continuously-differentiable interpolation can be used for sub-pixel resolution. We denote $D(\cdot)$ as the L_2 distance between a pair of pixel descriptors: $D(I_a,u_a,I_b,u_b) \triangleq ||f(I_a)(u_a) - f(I_b)(u_b)||_2$. At each iteration of training, a large number (on the order of 1 million total) of matches N_{matches} and non-matches $N_{\text{non-matches}}$ are generated between images I_a and I_b. The images are mapped to corresponding descriptor images via $f(\cdot)$ and the loss function is

$$\mathcal{L}_{\text{matches}}(I_a,I_b) = \frac{1}{N_{\text{matches}}} \sum_{N_{\text{matches}}} D(I_a,u_a,I_b,u_b)^2 \tag{1}$$

$$\mathcal{L}_{\text{non-matches}}(I_a,I_b) = \frac{1}{N_{\text{non-matches}}} \sum_{N_{\text{non-matches}}} \max(0, M - D(I_a,u_a,I_b,u_b))^2 \tag{2}$$

$$\mathcal{L}(I_a,I_b) = \mathcal{L}_{\text{matches}}(I_a,I_b) + \mathcal{L}_{\text{non-matches}}(I_a,I_b) \tag{3}$$

3.2 Training Procedures for Object-Centric Descriptors

Prior work [8] has used dynamic reconstruction [21] of raw RGBD data for only within-scene data association and remarkably showed that even without cross-scene data association, descriptors could be learned that were consistent across many dynamic scenes of the upper body of a human subject. While dynamic reconstruction is powerful, the challenges of topology changes [22] and difficulties of occlusion make it difficult to reliably deploy for an autonomous system. Schmidt et al. [8] also used data associations from static scene reconstructions for the task of relocalization in the same static environment. In contrast we sought to use only static reconstruction but seek consistency for dynamic objects. Other work [11] obtains dense descriptor consistency for a curated dataset of celebrity faces using only image warping for data association.

Using our robot mounted camera we are able to reliably collect high quality dense reconstructions for static scenes. Initially we applied only static-scene reconstruction to learn descriptors for specific objects, but we found that the learned object descriptors were not naturally consistent for challenging datasets with objects in significantly different configurations. Subsequently we developed techniques that leverage 3D reconstruction change detection, data augmentation, and loss function balancing to reliably produce consistent object representations with only static-scene data association for the wide variety of objects we have tested. These techniques also improve the precision of correspondences, as is discussed in Section 5.1. While we have tried many other ideas (see Appendix D.2), these are the techniques that were empirically found to significantly improve performance.

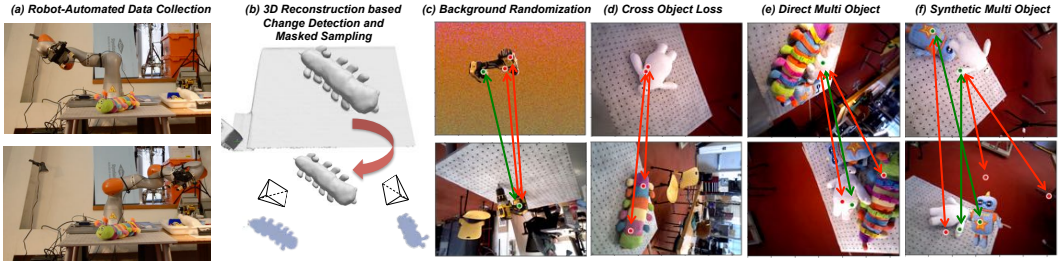


Figure 1: Overview of the data collection and training procedure. (a) automated collection with a robot arm. (b) change detection using the dense 3D reconstruction. (c)-(f) matches depicted in green, non-matches depicted in red.

Object masking via 3D change detection. Since we are trying to learn descriptors of objects that take up only a fraction of a full image, we observe significant improvements if the representational power of the models are focused on the objects rather than the backgrounds. A 640×480 image contains 307,200 pixels

but an image in our dataset may have as few as 1,000 to 10,000 of those pixels, or .3%-3%, that correspond to the object of interest. Initial testing with human-labeled object masks [23] showed that if matches for data associations were sampled only on the object (while non-matches were sampled from the full image) then correspondence performance was significantly improved. In order to provide autonomous object masking without any human input, we leverage our 3D reconstructions and results from the literature on 3D change detection [24] to recover the object-only part of the reconstruction (Figure 1b). Projecting this geometry into each camera frame yields object masks for each image. We want to emphasize that automatic object masking enables many other techniques in this paper, including: background domain randomization, cross-object loss, and synthetic multi-object scenes.

Background domain randomization. A strategy to encourage cross-scene consistency is to enforce that the learned descriptors are not reliant on the background. Since we have autonomously acquired object masks, we can domain randomize [25] the background (Figure 1c top) to encourage consistency – rather than memorizing the background (i.e. by describing the object by where it is relative to a table edge), the descriptors are forced to be representative of only the object.

Hard-negative scaling. Although as in [8] we originally normalized $\mathcal{L}_{\text{matches}}$ and $\mathcal{L}_{\text{non-matches}}$ by N_{matches} and $N_{\text{non-matches}}$, respectively, we found that what we call the "hard-negative rate", i.e. the percentage of sampled non-matches for which $M - D(I_a, u_a, I_b, u_b) > 0$ would quickly drop well below 1% during training. While not precisely hard-negative mining [26], we empirically measure improved performance if rather than scaling $\mathcal{L}_{\text{non-matches}}$ by $N_{\text{non-matches}}$, we adaptively scale by the number of hard negatives in the non-match sampling, $N_{\text{hard-negatives}}$, where $\mathbb{1}$ is the indicator function:

$$N_{\text{hard-negatives}} = \sum_{N_{\text{non-matches}}} \mathbb{1}(M - D(I_a, u_a, I_b, u_b) > 0) \tag{4}$$

$$\mathcal{L}_{\text{non-matches}}(I_a, I_b) = \frac{1}{N_{\text{hard-negatives}}} \sum_{N_{\text{non-matches}}} max(0, M - D(I_a, u_a, I_b, u_b))^2 \tag{5}$$

Data augmentation and normalization. While we collect only a modest number of scenes (4-10) per object or class, we ensure they are diverse in orientations, crops, and lighting conditions. We also applied synthetic 180-degree rotations randomly to our images. Additionally we find gains in performance by projecting all features to the unit sphere, i.e. $f(I)(u) \leftarrow \frac{f(I)(u)}{||f(I)(u)||}$ when using high-dimensional descriptors spaces (i.e., more than $D = 4$). This is explained further in Appendix D.1.

3.3 Multi-Object Dense Descriptors

We of course would like robots to have dense visual models of more than just one object. When we began this work it wasn't obvious to us what scale of changes to our training procedure or model architecture would be required in order to simultaneously (a) achieve individual single-object performance comparable to a single-object-only model, while also (b) learn dense visual descriptors for objects that are *globally distinct* – i.e., the bill of a hat would occupy a different place in descriptor space than the handle of a mug. To achieve distinctness, we introduce three strategies:

i. Cross-object loss. The most direct way to ensure that different objects occupy different subsets of descriptor space is to directly impose *cross-object loss* (Figure 1d). Between two different objects, we know that each and every pair of pixels between them is a non-match. Accordingly we randomly select two images of two different objects, randomly sample many pixels from each object (enabled by object masking), and apply non-match loss (with hard-negative scaling) to all of these pixel pairs.

ii. Direct training on multi-object scenes. A nice property of pixelwise contrastive loss, with data associations provided by 3D geometry, is that we can directly train on multi-object, cluttered scenes *without any individual object masks* (Figure 1e). This is in contrast with training pixelwise semantic segmentation, which requires labels for each individual object in clutter that may be difficult to attain, i.e. through human labeling. With pixel-level data associations provided instead by 3D geometry, the sampling of matches and the loss function still makes sense, even in clutter.

iii. Synthetic multi-object scenes. We can also synthetically create multi-object scenes by layering object masks [4]. To use dense data associations through synthetic image merging, we prune matches that become occluded during layering (Figure 1f). A benefit of this procedure is that we can create a combinatorial number of "multi-object" scenes from only single object-scenes, and can cover a wide range of occlusion types without collecting physical scenes for each.

4 Experimental

Data Collection and Pre-Processing. The minimum requirement for raw data is to collect an RGBD video of an object or objects. Figure 1 shows our experimental setup; we utilize a 7-DOF robot arm (Kuka IIWA LBR) with an RGBD sensor (Primesense Carmine 1.09) mounted at the end-effector. With the robot arm, data collection can be highly automated, and we can achieve reliable camera poses by using forward kinematics along with knowledge of the camera extrinsic calibration. For dense reconstruction we use TSDF fusion [27] of the depth images with camera poses provided by forward kinematics. An alternative route to collecting data which does not require a calibrated robot is to use a dense SLAM method (for example, [28, 29]). In between collecting RGBD videos, the object of interest should be moved to a variety of configurations, and the lighting can be changed if desired. While for many of our data collections a human moved the object between configurations, we have also implemented and demonstrated (see our video) the robot autonomously rearranging the objects, which highly automates the object learning process. We employ a Schunk two-finger gripper and plan grasps directly on the object point cloud (Appendix C). If multiple different objects are used, currently the human must still switch the objects for the robot and indicate which scenes correspond to which object, but even this information could be automated by the robot picking objects from an auxiliary bin.

Training Dense Descriptors. For training, at each iteration we randomly sample between some subset of specified image comparison types (Single Object Within Scene, Different Object Across Scene, Multi Object Within Scene, Synthetic Multi Object), and then sample some set of matches and non-matches for each. In this work, we use only static-scene reconstructions, so pixel matches between images can be easily found by raycasting and reprojecting against the dense 3D reconstruction model, and appropriately checking for occlusions and field-of-view constraints. For the dense descriptor mapping we train a 34-layer, stride-8 ResNet pretrained on ImageNet, but we expect any fully-convolutional network (FCN) that has shown effectiveness on semantic segmentation tasks to work well. Additional training details are contained in Appendix D.

5 Results

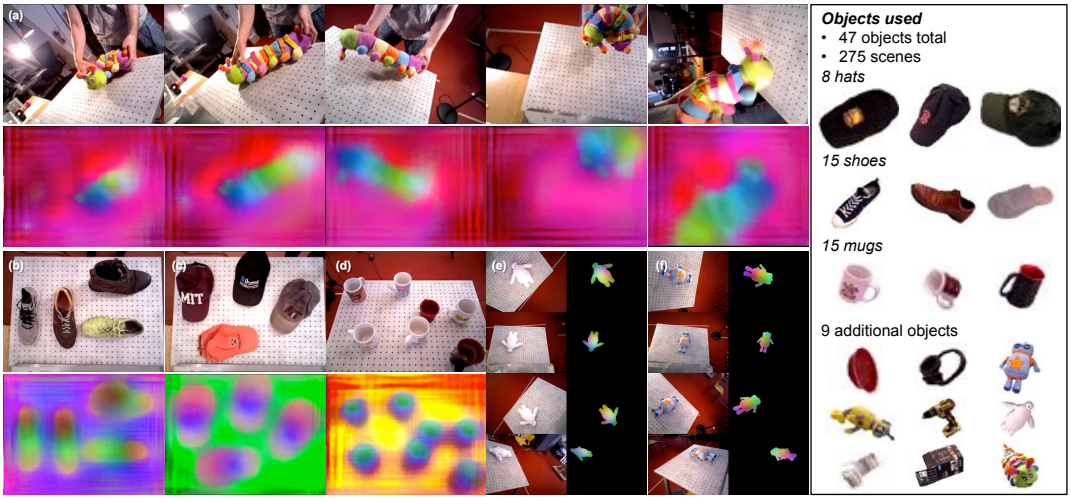


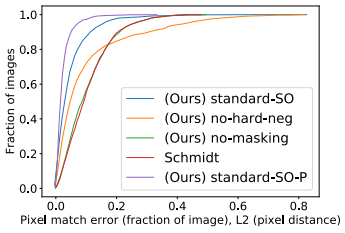
Figure 2: Learned object descriptors can be consistent across significant deformation (a) and, if desired, across object classes (b-d). Shown for each (a) and (b-d) are RGB frames (top) and corresponding descriptor images (bottom) that are the direct output of a feed-forward pass through a trained network. (e)-(f) shows that we can learn descriptors for low texture objects, with the descriptors masked for clear visualization. Our object set is also summarized (right).

5.1 Single-Object Dense Descriptors

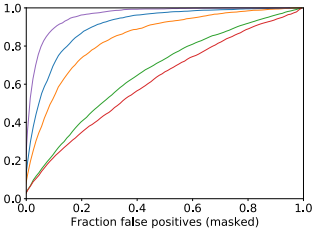
We observe that with our training procedures described in Section 3.2, for a wide variety of objects we can acquire dense descriptors that are invariant to viewpoint, configuration, and deformation. The variety of objects includes moderately deformable objects such as soft plush toys, shoes, mugs, and hats, and can include very low-texture objects (Figure 2). Many of these objects were just grabbed from around the lab (including the authors' and labmates' shoes and hats), and dense visual models can be reliably trained with

	single or multi object dataset	masked match sampling	scale by hard negatives	cross-object loss
standard-SO	single	✓	✓	
no-masking	single		✓	
no-hard-neg	single	✓		
Schmidt	single			
consistent	multi	✓	✓	
specific	multi	✓	✓	✓

(a)



(b)



(c)

Figure 3: (a) table describing the network training procedures referenced in experiments. (**standard-SO** = "standard single object". **standard-SO-P** is detailed in Appendix D.1). (b) Plots the cdf of the L2 pixel distance (normalized by image diagonal, 800 for a 640 x 480 image) between the best match $\hat{u}_b$ and the true match u_b^*, e.g. for **standard-SO** in 93% of image pairs the normalized pixel distance between u_b^* and $\hat{u}_b$ is less than 13%. All networks were trained on the same dataset. (c) Plots the cdf of the fraction of pixels u_b of the object pixels with $D(I_a,u_a^*,I_b,u_b) < D(I_a,u_a^*,I_b,u_b^*)$, i.e. they are closer in descriptor space to u_a^* than the true match u_b^*.

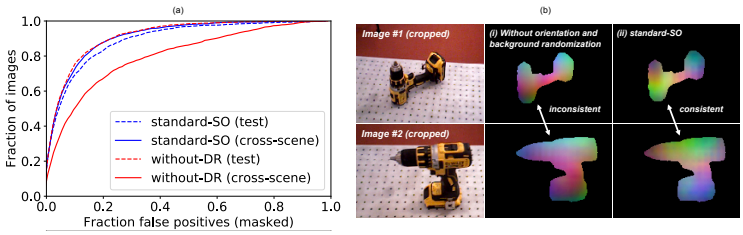


Figure 4: (a), with same axes as Figure 3b, compares **standard-SO** with **without-DR**, for which the only difference is that **without-DR** used no background domain randomization during training. The dataset used for (a) is of three objects, 4 scenes each. (b) shows that for a dataset containing 10 scenes of a drill, learned descriptors are inconsistent without background and orientation randomization during training (middle), but consistent with them (right).

the same network architecture and training parameters. The techniques in Section 3.2 provide significant improvement in both (a) qualitative consistency over a wide variety of viewpoints, and (b) quantitative precision in correspondences. As with other works that learn pairwise mappings to some descriptor space [30], in practice performance can widely vary based on specific sampling of data associations and non-associations used during training. One way to quantitatively evaluate correspondence precision is with human-labeled (used only for evaluation; never for training) correspondences across two images of an object in *different* configurations. Given two images I_a,I_b containing the same object and pixel locations $u_a^* \in I_a, u_b^* \in I_b$ corresponding to the same physical point on the object, we can use our dense descriptors to estimate u_b^* as $\hat{u}_b$:

$$\hat{u}_b \triangleq \underset{u_b \in I_b}{\operatorname{argmin}} D(I_a,u_a^*,I_b,u_b) \tag{6}$$

Figure 3 (b-c) shows a quantitative comparison of ablative experiments, for four different training procedures described in Figure 3a. Our new standard single-object training procedure (**standard-SO**) performs significantly better than our implementation of prior work's training procedures (**Schmidt**), and we isolate and measure significant improvement in correspondence precision for both object-masking and hard-negative scaling. We also find that for some low-texture objects, orientation randomization and background domain randomization are critical for attaining consistent object descriptors. Otherwise the model may learn to memorize which side of the object is closest the table, rather than a consistent object model (Figure 4b). Background domain randomization is most beneficial for smaller datasets, where it can significantly reduce overfitting and encourage consistency (Figure 4a); it is less critical for high-texture objects and larger datasets.

5.2 Multi-Object Dense Descriptors

An early observation during experimentation was that *overlap* in descriptor space naturally occurs if the same model is trained simultaneously on different singulated objects, where sampling of matches and non-matches was only performed *within scene*. Since there is no component of the loss function that requires different objects to occupy different subsets of descriptor space, the model maps them to an overlapping subset of descriptor space, distinct from the background but not each other (Figure 5a). Accordingly we sought to answer the question of whether or not we could *separate* these objects into unique parts of descriptor space.

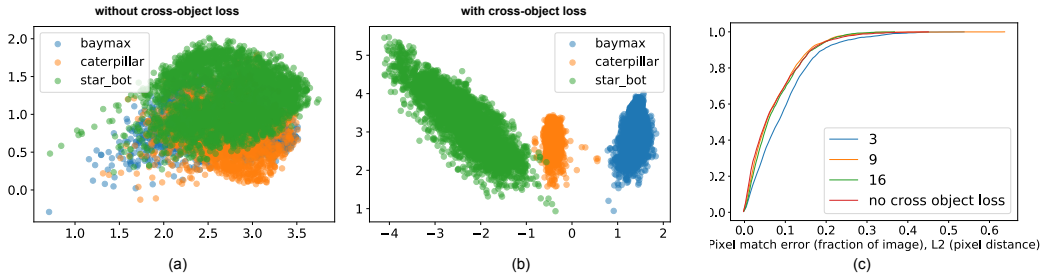


Figure 5: Comparison of training without any distinct object loss (a) vs. using cross-object loss (b). In (b), 50% of training iterations applied cross-object loss and 50% applied single-object within-scene loss, whereas (a) is 100% single-object within-scene loss. The plots show a scatter of the descriptors for 10,000 randomly-selected pixels for each of three distinct objects. Networks were trained with $D = 2$ to allow direct cluster visualization. (c) Same axes as Figure 3 (a). All networks were trained on the same 3 object dataset. Networks with a number label were trained with cross object loss and the number denotes the descriptor dimension. no-cross-object is a network trained without cross object loss.

By applying cross-object loss (Section 3.3.i, training mode **specific** in Figure 3a), we can convincingly separate multiple objects such that they each occupy distinct subsets of descriptor space (Figure 5b). Note that cross-object loss is an extension of sampling *across scene* as opposed to only *within scene*. Given that we can separate objects in descriptor space, we next investigate: does the introduction of object distinctness significantly limit the ability of the models to achieve correspondence precision for each individual object? For multi-object datasets, we observe that there is a measurable decrease in correspondence precision for small-dimensional descriptor spaces when the cross-object loss is introduced, but we can recover correspondence precision by training slightly larger-dimensional descriptor spaces (Figure 5c). For the most part, 3-dimensional descriptor spaces were sufficient to achieve saturated (did not improve with higher-dimension) correspondence precision for single objects, yet this is often not the case for distinct multi-object networks.

5.3 Selective Class Generalization or Instance Specificity

Surprisingly we find that when trained simultaneously on similar items of a class using training mode **consistent**, the learned descriptors naturally generalize well across sufficiently similar instances of the class. This result of converging descriptors across a class is similar to the surprising generalization observed for human datasets in [8, 11]. Here we show that we can obtain class consistent dense descriptors for 3 different classes of objects (hats, shoes, and mugs) trained with only static-scene data association. We observe that the descriptors are consistent despite considerable differences in color, texture, deformation, and even to some extent underlying shape. The training requirements are reasonably modest – only 6 instances of hats were used for training yet the descriptors generalize well to unseen hats, including a blue hat, a color never observed during training. The generalization extends to instances that a priori we thought would be failure modes: we expected the boot (Figure 6h) to be a failure mode but there is still reasonable consistency with other shoes. Sufficiently different objects are not well generalized, however – for example Baymax and Starbot (Figure 2e,f) are both anthropomorphic toys but we do not attain general descriptors for them. While initially we proposed research into further encouraging consistency within classes, for example by training a Dense Object Net to fool an instance-discriminator, the level of consistency that naturally emerges is remarkable and was sufficient for our desired levels of precision and applications.

For other applications, however, instance-specificity is desired. For example, what if you would like your robot to recognize a certain point on hat A as distinct from the comparable point on hat B? Although we could separate very distinct objects in multi-object settings as discussed in the previous section, it wasn't obvious to us if we could satisfactorily separate objects of the same class. We observe, however, that by applying the multi-object techniques (**specific** in Figure 3) previously discussed, we can indeed learn distinct descriptors even for very similar objects in a class (Figure 6iv).

5.4 Example Applications to Robotic Manipulation: Grasping Specific Points

Here we demonstrate a variety of manipulation applications in grasping specific points on objects, where the point of interest is specified in a reference image. We emphasize there could be many other applications, as mentioned in the Conclusion. In our demonstrations, a user clicks on just one pixel u_a^* in one reference

image. Now the robot has the ability to autonomously identify the corresponding point in new scenes via Equation 6. Akin to other works with similarity learning in metric spaces [30], we set a simple threshold to determine whether a valid match exists. If a match is identified in the new scene we can instruct the robot to autonomously grasp this point by looking up the corresponding location in the point cloud and using simple geometric grasping techniques (details in Appendix C).

The particular novel components of these manipulation demonstrations are in grasping the visual corresponding points for arbitrary pixels that are either in different (potentially deformed) configurations (Fig. 6i-ii), general across instances of classes (Fig. 6iii), or instance-specific in clutter (Fig. 6iv). Our video[1] best displays these tasks. Note that only a dense (as opposed to sparse) method can easily accommodate the arbitrary selection of interaction points, and class-generalization is out of scope for hand-designed descriptors such as SIFT. This is also out of scope for general grasp planners like [3, 18, 2, 20] which lack any visual object representation, and for segmentation based methods [18, 5, 4] since the visual representation provided by segmentation doesn't capture any information beyond the object mask.

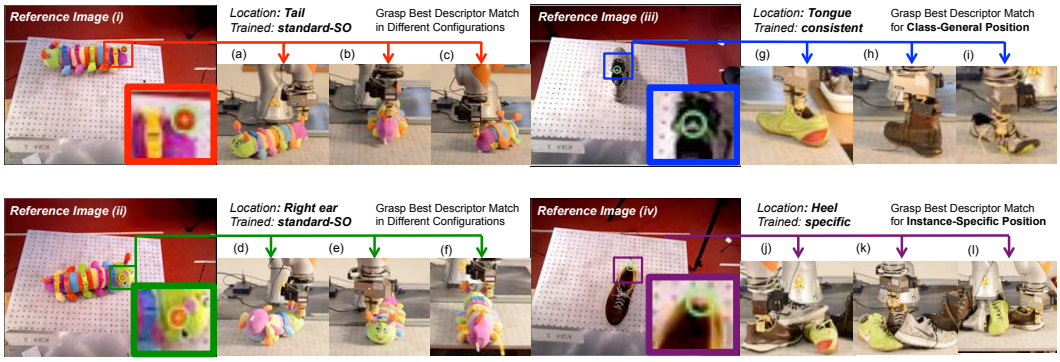


Figure 6: Depiction of "grasp specific point" demonstrations. For each the user specifies a pixel in a single reference image, and the robot automatically grasps the best match in test configurations. For single-object demonstrations, two different points for the caterpillar object are shown: tail (i) and right ear (ii). Note that the "right-ear" demonstration is an example of the ability to break symmetry on reasonably symmetrical objects. For class generalization (iii), trained with **consistent**, the robot grasps the class-general point on a variety of instances. This was trained on only 4 shoes and extends to unseen instances of the shoe class, for example (iii-i). For instance-specificity (iv) trained with **specific** and augmented with synthetic multi object scenes (3.3.iii), the robot grasps this point on the specific instance even in clutter.

6 Conclusion

This work introduces Dense Object Nets as visual object representations which are useful for robotic manipulation and can be acquired with only robot self-supervision. Building on prior work on learning pixel-level data associations we develop new techniques for object-centricness, multi-object distinct descriptors, and learning dense descriptors *by and for* robotic manipulation. Without these object centric techniques we found that data associations from static-scene reconstructions were not sufficient to achieve consistent object descriptors. Our approach has enabled automated and reliable descriptor learning at scale for a wide variety of objects (47 objects, and 3 classes). We also show how learned dense descriptors can be extended to the multi object setting. With new contrastive techniques we are able to train Dense Object Nets that map different objects to different parts of descriptor space. Quantitative experiments show we can train these multi object networks while still retaining the performance of networks that do not distinguish objects. We also can learn class-general descriptors which generalize across different object instances, and demonstrated this result for three classes: shoes, hats, and mugs. Using class-general descriptors we demonstrate a robot transferring grasps across different instances of a class. Finally we demonstrate that our distinct-object techniques work even for objects which belong to the same class. This is demonstrated by the robot grasping a specific point on a target shoe in a cluttered pile of shoes. We believe Dense Object Nets can enable many new approaches to robotic manipulation, and are a novel object representation that addresses goals (i-iv) stated in the abstract. In future work we are interested to explore new approaches to solving manipulation problems that exploit the dense visual information that learned dense descriptors provide, and how these dense descriptors can benefit other types of robot learning, e.g. learning how to grasp, manipulate and place a set of objects of interest.

[1] See video (`https://youtu.be/L5UW1VapKNE`) for extensive videos of the different types of robot picking.

Acknowledgments

The authors thank Duy-Nguyen Ta (calibration), Alex Alspach (hardware), Yunzhu Li (training techniques), Greg Izatt (robot software), and Pat Marion (perception and visualization) for their help. We also thank Tanner Schmidt for helpful comments in preparing the paper. This work was supported by: Army Research Office, Sponsor Award No. W911NF-15-1-0166; Draper Laboratory Incorporated, Sponsor Award No. SC001-0000001002; Lincoln Laboratory/Air Force, Sponsor Award No. 7000374874; Amazon Research Award, 2D-01029900; Lockheed Martin Corporation, Sponsor Award No. RPP2016-002. Views expressed in the paper are not endorsed by the sponsors.

References

[1] S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17(1):1334–1373, 2016.

[2] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. A. Ojea, and K. Goldberg. Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. *arXiv preprint arXiv:1703.09312*, 2017.

[3] M. Gualtieri, A. ten Pas, K. Saenko, and R. Platt. High precision grasp pose detection in dense clutter. In *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, pages 598–605. IEEE, 2016.

[4] M. Schwarz, C. Lenz, G. M. Garcıa, S. Koo, A. S. Periyasamy, M. Schreiber, and S. Behnke. Fast object learning and dual-arm coordination for cluttered stowing, picking, and packing. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[5] A. Milan, T. Pham, K. Vijay, D. Morrison, A. Tow, L. Liu, J. Erskine, R. Grinover, A. Gurman, T. Hunn, et al. Semantic segmentation from limited training data. *arXiv preprint arXiv:1709.07665*, 2017.

[6] E. Jang, S. Vijaynarasimhan, P. Pastor, J. Ibarz, and S. Levine. End-to-end learning of semantic grasping. *arXiv preprint arXiv:1707.01932*, 2017.

[7] C. B. Choy, J. Gwak, S. Savarese, and M. Chandraker. Universal correspondence network. In *Advances in Neural Information Processing Systems*, pages 2414–2422, 2016.

[8] T. Schmidt, R. Newcombe, and D. Fox. Self-supervised visual descriptor learning for dense correspondence. *IEEE Robotics and Automation Letters*, 2(2):420–427, 2017.

[9] C. Liu, J. Yuen, and A. Torralba. Sift flow: Dense correspondence across scenes and its applications. *IEEE transactions on pattern analysis and machine intelligence*, 33(5):978–994, 2011.

[10] R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1735–1742. IEEE, 2006.

[11] J. Thewlis, H. Bilen, and A. Vedaldi. Unsupervised learning of object frames by dense equivariant image labelling. In *Advances in Neural Information Processing Systems*, pages 844–855, 2017.

[12] A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, and T. Funkhouser. 3dmatch: Learning local geometric descriptors from rgb-d reconstructions. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 199–208. IEEE, 2017.

[13] J. Taylor, J. Shotton, T. Sharp, and A. Fitzgibbon. The vitruvian manifold: Inferring dense correspondences for one-shot human pose estimation. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 103–110. IEEE, 2012.

[14] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon. Scene coordinate regression forests for camera relocalization in rgb-d images. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 2930–2937. IEEE, 2013.

[15] E. Brachmann, A. Krull, F. Michel, S. Gumhold, J. Shotton, and C. Rother. Learning 6d object pose estimation using 3d object coordinates. In *European conference on computer vision*, pages 536–551. Springer, 2014.

[16] C. Finn and S. Levine. Deep visual foresight for planning robot motion. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 2786–2793. IEEE, 2017.

[17] A. Nair, D. Chen, P. Agrawal, P. Isola, P. Abbeel, J. Malik, and S. Levine. Combining self-supervised learning and imitation for vision-based rope manipulation. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 2146–2153. IEEE, 2017.

[18] A. Zeng, S. Song, K.-T. Yu, E. Donlon, F. R. Hogan, M. Bauza, D. Ma, O. Taylor, M. Liu, E. Romo, et al. Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching. *arXiv preprint arXiv:1710.01330*, 2017.

[19] A. Zeng, S. Song, S. Welker, J. Lee, A. Rodriguez, and T. Funkhouser. Learning synergies between pushing and grasping with self-supervised deep reinforcement learning. *arXiv preprint arXiv:1803.09956*, 2018.

[20] L. Pinto and A. Gupta. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pages 3406–3413. IEEE, 2016.

[21] R. A. Newcombe, D. Fox, and S. M. Seitz. Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 343–352, 2015.

[22] W. Gao and R. Tedrake. Surfelwarp: Efficient non-volumetric single view dynamic reconstruction. Robotics: Science and Systems, 2018.

[23] P. Marion, P. R. Florence, L. Manuelli, and R. Tedrake. Labelfusion: A pipeline for generating ground truth labels for real rgbd data of cluttered scenes. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[24] R. Finman, T. Whelan, M. Kaess, and J. J. Leonard. Toward lifelong object segmentation from change detection in dense rgb-d maps. In *Mobile Robots (ECMR), 2013 European Conference on*, pages 178–185. IEEE, 2013.

[25] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*, pages 23–30. IEEE, 2017.

[26] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1627–1645, 2010.

[27] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 303–312. ACM, 1996.

[28] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on*, pages 127–136. IEEE, 2011.

[29] T. Whelan, S. Leutenegger, R. Salas-Moreno, B. Glocker, and A. Davison. Elasticfusion: Dense slam without a pose graph. Robotics: Science and Systems, 2015.

[30] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

(a) **(b)**

Figure 7: (a) Kuka IIWA LRB robot arm. (b) Schunk WSG 50 gripper with Primesense Carmine 1.09 attached

Appendix A Experimental Hardware

All of our data was collected using an RGBD camera mounted on the end effector of a 7 DOF robot arm (see Figure 7). We used a Kuka IIWA LBR robot with a Schunk WSG 50 parallel jaw gripper. A Primesense Carmine 1.09 RGBD sensor was mounted to the Schunk gripper and precisely calibrated for both intrinsics and extrinsics.

Appendix B Experimental Setup: Data Collection and Pre-Processing

As discussed in the paper all of our data consists of 3D dense reconstructions of a static scene. To collect data for a single scene we place an object (or set of objects) on a table in front of the robot. We then perform a scanning pattern with the robot which allows the camera to capture many different viewpoints of the static scene. This procedure takes about 70 seconds during which approximately 2100 RGBD frames are captured (the sensor outputs at 30Hz). Using the forward kinematics of the robot, together with our camera extrinsic calibration, we also record the precise camera pose corresponding to each image. Since we have the camera poses corresponding to each depth image we use TSDF fusion [27] to obtain a dense 3D reconstruction. Although we use our robot's forward kinematics to produce the dense 3D reconstruction together with camera pose tracking, any dense SLAM method (such as [28]) could be used instead. In practice we found that using the robot's forward kinematics to obtain camera pose estimates produces very reliable 3D reconstructions which are robust to lack of geometric or RGB texture, varying lighting conditions, etc. Next we obtain new depth images for each recorded camera frame by rendering against the 3D reconstruction using our camera pose estimates. This step produces depth images which are globally consistent and free of artifacts (i.e. missing depth values, noise etc.). This keeps the process of finding correspondences across RGB images as a simple operation between poses and depth images. To enable the specific training techniques discussed in the paper we also need to know which parts of the scene correspond to the objects of interest. To do this we implemented the change detection technique of [24]. In practice since all of our data was collected on a tabletop, and our reconstructions can be easily globally aligned (due to the fact that we know the global camera poses from the robot's forward kinematics) we can simply crop the reconstruction to the area above the table. Once we have isolated the part of the reconstruction corresponding to the objects of interest, we can easily render binary masks via the same procedure as was used to generate the depth images.

Our RGBD sensor captures images at 30Hz, so we downsample the images to avoid having images which are too visually similar. Specifically we downsample images so that the camera poses are sufficiently different (at least 5cm of translation, or 10 degrees of rotation). After downsampling we are left with approximately 315 images per scene.

In between collecting RGBD videos, the object of interest should be moved to a variety of configurations, and the lighting can be changed if desired. While for many of our data collections a human moved the object between configurations, we have also implemented and demonstrated (see our video) the robot autonomously rearranging the objects, which highly automates the object learning process. We employ a Schunk two-finger gripper and plan grasps directly on the object point cloud (see Appendix C). If multiple different objects are used, currently the human must still switch the objects for the robot and indicate which

scenes correspond to which object, but even this information could be automated by the robot picking objects from an auxiliary bin and use continuity to simply identify which logs are of the same object.

Appendix C Grasping Pipeline

While our learned visual descriptors can help us determine *where* to grasp, they can't be used during the bootstrapping phase before visual learning has occurred, and they don't constrain the 6DOF orientation of the gripper. Accordingly to choose grasps our pipeline employs simple geometric point cloud based techniques. There are two types of grasping that are performed in the paper. The first is performed while grasping the object to automatically reorient it during data collection, during the visual learning process. To achieve this we first use the depth images and camera poses to fuse a point cloud of the scene. We then randomly sample many grasps on the point cloud and prune those that are in collision. The remaining collision free grasps are then scored by a simple geometric criterion that evaluates the grasps for antipodality. The highest scoring grasp is then chosen for execution. The second type of grasping is, as in Section 5.4 of the paper, when we are attempting to grasp a specific point. In this case the robot moves to a handful of pre-specified poses and records those RGBD images. The RGB images are used to look up the best descriptor match and determine a pixel space location to grasp in one image, and the corresponding depth image and camera pose are used to determine the 3D location of this point. The handful of depth images are also fused into a point cloud, and the grasping procedure is almost the same as the first type, with the slight modification that all the randomly sampled grasps are centered around the target grasp point. Although there are a variety of learning based grasping algorithms [3, 2] that could have been used, we found that our simple geometric based grasp planning was sufficient for the tasks at hand.

Appendix D Network Architecture and Training Details

For our network we use 34-layer, stride-8 ResNet (pretrained on ImageNet), and then bilinearly upsample to produce a full resolution $640x480$ image.

For training, at each iteration we randomly sample between some specified subset of specified image comparison types (Single Object Within Scene, Different Object Across Scene, Multi Object Within Scene, Synthetic Multi Object). The weighting between how often each type is chosen is done via specifying their probabilities of being selected. Once the type has been sampled we then sample some set of matches and non-matches for each (around 1 million in total). Each step of the optimizer uses a single pair of images.

All the networks were trained with the same optimizer settings. Networks were trained for 3500 steps using an Adam optimizer with a weight decay of $1e-4$. Training was performed on a single Nvidia 1080 Ti or Titan Xp, a single step takes approximately 0.22 seconds, i.e. approximately 13 minutes, and so together with collecting a handful of scenes the entire training for a new object can take 20 minutes. The learning rate was set to $1e-4$ and dropped by 0.9 every 250 iterations. The networks trained with procedure **specific** used a 50-50 split of within scene image pairs and across scene image pairs 50% of the time. For the network used to grasp the heel of the red/brown shoe in Section 5.4 we sampled equally the three data types (Single Object Within Scene, Different Object Across Scene, Synthetic Multi Object).

D.1 Descriptor Projection to Unit Sphere

There is one additional small feature we discovered prior to camera-ready submission that gives substantial quantitative performance gains, although it was not used nor needed for most of the experiments including all hardware experiments. As is standard in many metric learning works, for example [30], we can add a simple parameterless normalization layer in which we project all features to the unit sphere, $f(I)(u) \leftarrow \frac{f(I)(u)}{||f(I)(u)||}$. This is contrast to channel-wise normalization mentioned in [7]. Given that there is a projection to the unit-sphere manifold, higher dimensions are needed in order to see improvements. While without unit-sphere projection, we see saturation of performance at around $D=3$ for single-object descriptors, with unit-sphere projection we see significant gains in going from $D=4$ to $D=8$ even for single-object descriptors. Further work could investigate the effect of unit-sphere projection on descriptor consistency across classes. The network marked as **standard-SO-P** in Figures 3b and 3c was trained identically to **standard-SO**, but with the unit sphere projection, and with $D=16$.

D.2 Additional Approaches Which Did not Improve Performance

During the course of our experimentation we found that the network architecture and training procedure outlined in Section 3 gave the best performance. However we also tried a variety of other network architectures and loss functions which did not improve performance. We discuss a few of them here.

Triplet loss [30] uses a triplet loss instead of the contrastive loss. We implemented a pixelwise version of this triplet loss, but found that it actually reduced performance on our dataset.

Scaling loss by pixel distance Our loss function tries to ensure that non-matches have descriptors that are at least a margin M apart. It can be hard, however, for the network to take two pixels that are next to each other, and assign them significantly different descriptors. In an effort to try to less heavily penalize non-matches that are close to the true match, we introduced an additional scaling term using the L2 pixel distance. Following the notation from Section 3 let $u_a \in I_a, u_b \in I_b$ correspond to a non-match. Additionally, using the notation of Equation 6 let $u_b^* \in I_b$ correspond to the true match for u_a. Define $\Delta(u_b, u_b^*) \in \mathbb{R}$ to be the L2 distance, measured in pixels, between u_b and u_b^*. The scaled non-match loss is now defined as

$$\mathcal{L}_{\text{non-matches}}(I_a, I_b) = \frac{1}{N_{\text{non-matches}}} \sum_{N_{\text{non-matches}}} \frac{1}{M_p} \min(\Delta(u_b, u_b^*), M_p) \cdot \max(0, M - D(I_a, u_a, I_b, u_b))^2 \quad (7)$$

where M_p is the pixel distance at which this additional loss component saturates. Parameter sweeps on M_p, where $M_p = 1$ is the original un-scaled loss function, did not show significant differences in performance. A potentially useful extension for future work would be to try scaling by the geodesic distance in the 3D reconstructios.

Convolutional spatial transformer: We implemented a convolutional spatial transformer, as described in [7]. The convolutional spatial transformer is meant to help the network achieve scale and rotation invariance for each feature. We did not find any significant performance gains with our implementation of such a convolutional spatial transformer. A hypothesis for why we did not see performance gains is that our used network architecture (34-layer ResNet) was significantly deeper than the architecture used in [7], and our data collection and augmentation provided significant variety in scale and rotation – accordingly, our network must approximate scale and rotation invariance in order to fit the training data.

Ratios for sampling non-matches: Given a pixel u_a corresponding to a point on an object, non-matches u_b can either be on the object, or on the background. Let $I_{b,object}$ denote object pixels in image I_b, and $I_{b,background}$ denote background pixels. During training we experimented with varying the fraction of non-matches u_b which lie in $I_{b,object}$ vs. $I_{b,background}$. In general we found performance was insensitive to this ratio as long as the fraction of non-matches in $I_{b,object}$ was between 25% to 75%.