Discovery of Semantic 3D Keypoints via End-to-end Geometric Reasoning

Supasorn Suwajanakorn [1] **Noah Snavely** [1] **Jonathan Tompson** [1] **Mohammad Norouzi** [1]

Abstract

Given images of different objects from a single category, e.g. chairs, humans can easily identify a set of semantically-similar keypoints such as the legs, chair back, or arm rests in each image, as well as reason about their 3D relationship. However, this task is challenging for current vision algorithms because the appearance of the same semantic keypoint varies greatly across viewing angles and object instances. In addition, ground-truth correspondence is difficult to obtain at scale for learning-based approaches, and there is no objective and consistent metric that captures the optimal selection of keypoints. Unlike prior techniques that solve for 2D correspondence, which fail to model visual occlusion, we propose the first unsupervised technique that automatically discovers a semantically-consistent set of latent 3D keypoints across all objects of a particular class. We introduce a network architecture which achieves equivariance by design and we incorporate novel losses, including 3D consistency loss and Procrustes loss, which together help infer locations and depths. Our proposed network can also differentiate visually ambiguous parts of partially symmetric objects, such as cars, and predict keypoints that are completely occluded due to 180° change in viewpoint without relying on any explicit location prior.

1. Introduction

Past generations of object recognition models comprised hand-crafted feature extraction (*e.g.,* SIFT (Lowe, 1999) and HOG (Dalal & Triggs, 2005)), followed by classification. Convolutional neural networks (LeCun et al., 1998) present an alternative approach, in which the feature representation and the classification modules are jointly optimized. This end-to-end optimization achieves impressive results on object recognition (Krizhevsky et al., 2012; He et al., 2016) and segmentation (Long et al., 2015; He et al., 2017). In this paper, we examine whether one can obtain similar benefits from an end-to-end optimization of representations based on a *sparse* set of keypoints or landmarks, often used in geometric reasoning applications, such as 3D reconstruction (Snavely et al., 2006) and 3D shape retrieval.

For instance, consider the problem of determining the 3D pose of an object in an image. A typical approach often starts by detecting sparse, semantically meaningful keypoints, using keypoint detectors learned via explicit supervision. The training data for this supervision typically comes in the form of manual annotations of a set of representative keypoints in different images of a single object category. For instance, researchers have compiled keypoint annotations for large datasets of faces (Sagonas et al., 2016), hands (Tompson et al.), and human bodies (Andriluka et al., 2014; Lin et al., 2014). From such data, one learns a supervised mapping from images to keypoint positions, followed by geometric reasoning (*e.g.,*via a PnP algorithm (Lepetit et al., 2008)) on the sparse set of keypoint detections to recover the 3D pose or the camera angles from a given image. Such consistent annotation of sparse keypoints in images of different types of objects *e.g.,* cars, chairs, and airplanes is not only expensive, but is also ill-defined. To devise a reasonable set of points, one should take into account the downstream task. Moreover, not all keypoints are equally easy to detect; the ease of detectability of keypoints in different views of the objects plays an important role in the performance of any downstream task.

In this work, we propose a framework for learning these keypoints automatically, along with their detectors, by optimizing them for a specific downstream task, such as 3D pose estimation. Further, we advocate the use of 3D coordinates for keypoint positions, *i.e.,* we not only detect x and y coordinates for any keypoint, but also predict the corresponding depth. Confirming the recent developments on depth estimation from a single image, we find that detecting the depth is possible, particularly when the keypoints are detected jointly. One benefit of using 3D coordinates is that we can even detect invisible keypoints. More importantly, using 3D keypoints from a single view, one can reason about the 3D pose and the 3D shape of the object.

[1]Google. Correspondence to: Supasorn Suwajanakorn <supasorn@google.com>, Noah Snavely <snavely@google.com>, Jonathan Tompson <tompson@google.com>, Mohammad Norouzi <mnorouzi@google.com>.

2. Related Work

Both 2D and 3D keypoint detection are long-standing problems in the computer vision community, where keypoint inference is traditionally used as an early stage in object localization pipelines. As an example, a successful early application of modern CNNs was on 2D human-joint detection from monocular RGB images. Due to it‘s compelling utility for HCI, motion capture and security applications, a large body of work has since developed in this domain (Toshev & Szegedy, 2014; Tompson et al., 2014; Pishchulin et al., 2016; Newell et al., 2016; Yang et al., 2017; Papandreou et al., 2017; Huang et al., 2017; He et al., 2017).

More related to our work, a number of recent CNN-based techniques have been developed for 3D human-keypoint detection from monocular RGB images, which use varying architectures, supervised objectives, and 3D structural priors to directly infer a predefined set of 3D joint locations (Mehta et al., 2017c;a; Chen et al., 2017; Mehta et al., 2017b; Güler et al., 2018). Other techniques use inferred 2D keypoint detectors and learned 3D priors to perform “2D-to-3D-lifting” (Ramakrishna et al., 2012; Chen & Ramanan, 2017; Zhou et al., 2016b; Martinez et al., 2017). While in this work we similarly use a feed-forward CNN to predict 3D keypoints, our proposed system is different in two distinct ways. First, we predict keypoints on classes of objects with larger shape variation and less geometric consistency (for instance between car or chair instances with vastly different topologies). Second, the set of keypoints are not defined *a priori* and are instead a latent set that is optimized to improve inference performance for the 3D rigid pose regression problem.

A large body of work also exists for supervised keypoint detection on more generalized object classes. For instance, Hejrati & Ramanan (2012) propose a two stage model which first uses a modified deformable part model to reason about 2D shape and appearance variation, and then a second stage to directly reason about 3D shape and camera viewpoint. Their model learns robust 3D geometric consistency priors and is able to localize annotated keypoint positions for fully-occluded keypoints. Wu et al. (2016) devise an end-to-end framework for jointly predicting 2D keypoint heatmaps on real-world RGB images and 3D object structure. They define a novel “projection-layer” to project 3D synthetic structure onto supervised 2D correspondences and show that the resultant 3D keypoints can be used for a variety of common tasks.

While curating large-scale 3D annotated pose datasets (from in-the-wild RGB images) is notoriously difficult, time-consuming and costly, (Xiang et al., 2014) propose a dataset, *PASCAL3D+* which augments 12 object categories from PASCAL VOC with 3D annotations, including 3D camera distance and object pose. The authors use 3D CAD models and hand-crafted 3D and 2D keypoint locations to fit the models to monocular images.

Enforcing latent structure in CNN feature representations has been explored for a number of domains. For instance, Sabour et al. (2017) improve upon the “capsules” framework (Hinton et al., 2011), which encodes activation properties in the magnitude and direction of hidden-state vectors and then combines them to build higher-level features, by incorporating a novel routing-by-agreement algorithm to determine the connection of lower-level capsules to those higher in the network. The output of the CNN of this work can be seen as a similar form of latent 3D feature, whose position is encouraged to represent a 3D keypoint due to the carefully constructed consistency and relative pose terms of our objective function.

A number of recent publications have demonstrated 2D correspondence matching across intra-class instances with large shape and appearance variation. Choy et al. (2016) propose a fully-convolutional architecture and a novel appearance-based contrastive loss function to encode geometry and semantic similarity. Han et al. (2017) propose a novel SCNet architecture for learning a geometrically plausible model for 2D semantic correspondence. In this work, region proposals are used for initial match primitives and geometric consistency is incorporated directly in the SCNet loss function.

Thewlis et al. (2017) use ground-truth transforms (in their case optical flow between image pairs) and point-wise matching to learn a model which can infer a dense object-centric coordinate frame with viewpoint and image deformation invariance. Similarly, Agrawal et al. (2015) use egomotion and camera transformation prediction between image pairs to learn semi-supervised feature representations, and show that these features are competitive with supervised features for a variety of tasks. Arie-Nachimson & Basri (2009) propose a system that builds 3D models of classes of rigid objects and exploits these models to estimate 3D pose from monocular RGB. By factorizing known transformations between views, this model outputs a collection of latent features and their 3D locations, appearances in different views, and visibility properties. Inspired by successful usage of cycle consistency for learning correspondence (Huang & Guibas, 2013; Zhou et al., 2015), Zhou et al. (2016a) train a CNN to predict cross-instance correspondence between different objects of the same semantic class. At training time a synthetic CAD model is fit offline to both views to impose cycle consistency and then at test time no CAD models are required. Similarly, in our work we make use of a ground-truth transformation between camera views and a Euclidean loss on transformed points to encourage latent points to track geometric keypoints. However unlike this prior work, our keypoints are entirely in 3D and we incorporate multiple loss terms to improve relative

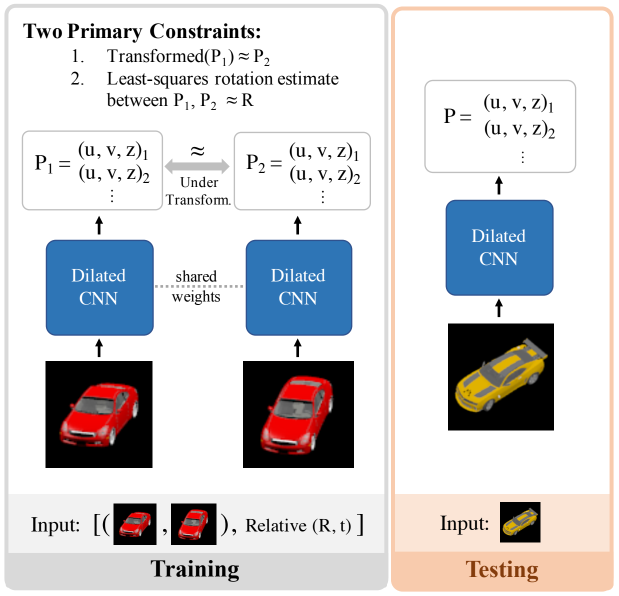
Two Primary Constraints:
1. Transformed(P1) ≈ P2
2. Least-squares rotation estimate between P1, P2 ≈ R
P1 = (u, v, z)1 (u, v, z)2 ⋮
≈ Under Transform.
P2 = (u, v, z)1 (u, v, z)2 ⋮
Dilated CNN
shared weights
Dilated CNN
Input: [(,), Relative (R, t)]
Training
P = (u, v, z)1 (u, v, z)2 ⋮
Dilated CNN
Input:
Testing

Figure 1. During training, we are provided with two views of the same object and their corresponding rigid transformation. We learn a set of 3D keypoints that are consistent in both views and enable the recovery of the transformation. During test, given a single input image, we predict the 3D coordinates of the keypoints.

rotation estimates and to encourage spatial diversity.

Automated approaches for 3D correspondence search have been recently investigated. Salti et al. (2015) cast 3D keypoint detection as a binary classification between points whos ground-truth similarity label is determined by a predefined 3D descriptor. This general descriptor allows their architecture to find correspondences across object classes with varying topology. As with this work, Zhou et al. (2017) use view-consistency as a supervisory signal on depth maps to perform 3D keypoint prediction. However in their work they use it as a regularization mechanism for keypoint prediction on unlabeled instances with intra-class variation, and additionally the authors demonstrate their approach on depth images only. Similarly, Su et al. (2015) leverage viewpoint matching and a corpus of synthetic computer graphics models to perform object viewpoint estimation. Their analysis-by-synthesis algorithm optimizes the viewpoint of synthetically rendered images to match the real-world RGB image viewpoint via a CNN viewpoint embedding.

3. End-to-end optimization of 3D keypoints

Given a single image of a known object category, our goal is to predict an ordered set of 3D keypoints, defined as pixel coordinates and associated depth values. Such keypoints are required to be semantically consistent across different viewing angles, as well as across different instances of the same object category. Some example keypoints for cars, chairs, and planes are included in Figure 4. These keypoints serve as the building blocks of feature representations based on a sparse set of points, useful for geometic reasoning and pose-aware object recognition (*e.g.,* Sabour et al. (2017)).

Rather than learning a supervised mapping from images to annotated keypoint positions, we do not *a priori* define the keypoint positions, and instead optimize them jointly with respect to a downstream task. In this paper, we focus on the task of *relative pose estimation*, where given two views of the same object, we would like to estimate the relative rigid transformation that relates one view to the other. Figure 1 gives an overview of the proposed technique. During training, we are given two views of the same object with a known rigid transformation T. Given two sets of N 3D keypoint positions detected in the two views (denoted P and P'), we formulate an objective function $O(P, P')$, based on which one can optimize a parametric mapping of images into these sets of keypoints. Our objective consists of two key components:

- A *multi-view consistency* loss that measures the discrepancy between the two sets of points under the ground truth transformation.
- A *relative pose estimation* objective, which penalizes the angular difference between the rotation $\hat{R}$ recovered from P and P' *vs.* the ground truth rotation R.

We demonstrate that by using view consistency and pose estimation in the objective, the model is able to discover important keypoints that seem natural for different object classes. In what follows, we first review the components of the objective function and then describe the architecture of our *keypoint prediction network*.

Notation. Each training tuple given to our algorithm comprises a pair of images (I, I') of the same object from different viewpoints and their relative rigid transformation $T \in SE(3)$ that converts the underlying 3D shape of I to I' with the following matrix form:

$$T = \begin{bmatrix} R^{3\times3} & t^{3\times1} \\ 0 & 1 \end{bmatrix}, \tag{1}$$

where R and t represent the rotation and translation respectively. We learn a parametric function $f_\theta(I)$ that maps a 2D image I to a set of 3D points $P = \{p_i \equiv (u_i, v_i, z_i)\}_{i=1}^N$. We optimize the parameters of this mapping denoted θ by gradient descent on an objective of the form $O(f_\theta(I), f_\theta(I'))$.

3.1. Multi-view consistency

For an individual 3D keypoint to track the same object part, we need to ensure that the keypoints are consistent across different views, *i.e.,*that the predicted 2D locations and depths are geometrically consistent with the known relative pose. This is the goal of our multi-view consistency loss. Specifically, a 3D keypoint in one image should project onto the same pixel location as the corresponding keypoint in the second image. For this task, we assume a perspective camera model with a known global focal length f. Below, we use $[x, y, z]$ to denote 3D coordinates, and $[u, v]$ to denote pixel coordinates. The projection of a keypoint $[u, v, z]$ from image I into image I' (and vice versa) is given by the projection operators:

$$\begin{aligned}[\hat{u'}, \hat{v'}, \hat{z'}, 1]^\top &\sim \pi T \pi^{-1}([u, v, z, 1]^\top) \\ [\hat{u}, \hat{v}, \hat{z}, 1]^\top &\sim \pi T^{-1} \pi^{-1}([u', v', z', 1]^\top)\end{aligned}$$

where for instance, $\hat{u'}$ denotes the projection of u to the second view, and $\hat{u}$ denotes the projection of u' to the first view. Here, $\pi : \mathbb{R}^4 \rightarrow \mathbb{R}^4$ represents the perspective projection operation that maps an input homogeneous 3D coordinate $[x, y, z, 1]^\top$ in camera coordinates to a pixel position plus depth:

$$\pi([x, y, z, 1]^\top) = \left[\frac{xf}{z}, \frac{yf}{z}, z, 1\right]^\top = [u, v, z, 1]^\top$$

We define a symmetric multi-view consistency loss as:

$$L_{\text{con}} = \frac{1}{2N} \sum_{i=1}^{N} \left\| [u_i, v_i, u'_i, v'_i]^\top - [\hat{u}_i, \hat{v}_i, \hat{u'_i}, \hat{v'_i}]^\top \right\|^2$$

We measure error only in the observable image space (u, v) as opposed to also using z, because depth is never directly observed, and usually has different units compared to u and v. Note however that predicting z is critical for us to be able to project points between the two views.

Enforcing multi-view consistency is sufficient to encourage the neural network to infer a consistent set of 2D keypoint positions (and depths) across different views. However, we found that selecting the keypoints only by minimizing a consistency loss often leads to a degenerate solution where all keypoints collapse to the same 3D point, *e.g.,* the center of mass of the object, which is not useful for most downstream tasks. One can encode an explicit notion of diversity to prevent degenerate solutions, but we find that including the downstream objective as part of the keypoint selection process naturally encourages keypoint separation. Further, there are infinitely many sets of keypoints that satisfy multi-view consistency; we need a clear downstream application to decide which keypoints are most useful.

3.2. Relative pose estimation

One important application of keypoint detection is to recover the relative transformation between a given pair of images. Accordingly, we define a differentiable objective that given two sets of detected keypoints, estimates the misfit between the estimation of the relative angle $\hat{R}$ (computed via Procrustes' alignment of the detected keypoints) and the ground truth angle R. Given the translation equivariance built into our keypoint prediction network and the view consistency loss above, we did not separately include the estimation error of the translation vector in the objective. Empirically, we find that the pose estimation objective helps significantly in producing a reasonable and natural selection of keypoints, leading to the automatic discovery of interesting parts such as each wheel of a car, the cockpit and wings of a plane, or the legs and back of a chair.

The pose estimation objective seeks two sets of keypoints that are optimal in the least-squares sense for recovering the relative transformation between the given pair of images. We define this loss as

$$L_{\text{pose}} = 2 \arcsin \left(\left\| \hat{R} - R \right\|_F / (2\sqrt{2}) \right)$$

which measures the angular distance between the optimal least-squares estimate $\hat{R}$ computed from the two sets of keypoints, and the ground truth relative rotation matrix R. Fortunately, we can formulate this objective in terms of fully differentiable operations.

To estimate $\hat{R}$, let X and $X' \in \mathbb{R}^{3\times N}$ denote two matrices comprising unprojected 3D keypoint coordinates for the two views. Let $X \equiv [X_1, \ldots, X_N]$ and

$$X_i \equiv (\pi^{-1} p_i)[:3] \tag{2}$$

where [:3] is the de-homogenize function (returns the first 3 coordinates). Similarly X' denotes unprojected points in P'. Let $\tilde{X}$ and $\tilde{X}'$ denote the mean-subtracted version of X and X', respectively. The optimal least-squares rotation $\hat{R}$ between the two sets of keypoints is then given by:

$$\hat{R} = V \operatorname{diag}(1, 1, \ldots, \det(VU^\top))U^\top$$

where:

$$U, \Sigma, V^\top = \operatorname{SVD}(\tilde{X}\tilde{X}'^\top)$$

This estimation problem to recover $\hat{R}$ is known as the orthogonal Procrustes problem (Schönemann, 1966). To ensure that $\tilde{X}\tilde{X'}^\top$ is invertible and to increase the robustness of the keypoints, we add a Gaussian noise to the 3D coordinates of the keypoints (X and X') and instead seek the best rotation under some noisy predictions of keypoints.

Note that there are other objectives that can lead to the discovery of interesting sets of keypoints. For example, one could find the optimal set of keypoints that best reconstruct the original input image under a specific decoder network. This reconstruction loss is explored in Transforming Auto-encoders (Hinton et al., 2011), although under a different context and applications. In this work we are interested in recovering the geometric characteristics of a scene, and accordingly we propose the relative pose estimation objective.

3.3. Additional keypoint characteristics

In addition to the main objectives introduced above, there are common and preferable characteristics of keypoints that lead to finding higher quality generic keypoints. We expect the following characteristics to benefit many possible downstream tasks:

- No two keypoints predicted in an image should share the same 3D location.
- Keypoints should lie within the object's silhouette.

Separation loss. To prevent two keypoints from occupying the same location, we introduce a term that penalizes two keypoints if they are closer than a hyperparameter δ in 3D,

$$L_{\text{sep}} = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j \neq i}^{N} \max\left(0, \delta^2 - \|X_i - X_j\|^2\right)$$

Unlike the consistency loss, this loss is computed in 3D space to allow multiple keypoints to occupy the same pixel location as long as they have different associated depths. Note that we prefer this form of a robust loss with a bounded support over an unbounded loss (*e.g.,* exponential discounting) because it does not exhibit a structure bias. In other words, it does not prefer certain keypoint configurations (*e.g.,* placing points on a honeycomb or infinitely far apart) as long as they are sufficiently far from one another.

Silhouette consistency. We encourage the keypoints to lie within the silhouette of the object of interest. Our network predicts the (u_i, v_i) cooridnates of the ith keypoint by predicting a spatial distribution over possible keypoint positions denoted $g_i(u, v)$. One way to ensure silhouette consistency, is by *only* allowing a non-zero probability distribution inside the silhouette of the object, as well as encouraging the spatial distribution to be concentrated, *i.e.,* uni-modal with a low variance.

During training, we have access to the binary segmentation mask of the object $b(u, v) \in \{0, 1\}$ in each image. The value of $b(u, v)$ is 1 if and only if this the pixel (u, v) belongs to the foreground object. The following term encourages the keypoints to lie within the silhouette mask:

$$L_{\text{obj}} = \frac{1}{N} \sum_{i=1}^{N} -\log \sum_{u,v} b(u, v) g_i(u, v)$$

Note that this binary flag is only part of the loss and not used at test time. Note that this objective incurs a zero cost if all of the probability mass lies within the silhouette. We also include a term to minimize the variance of each of the distribution maps:

$$L_{\text{var}} = \frac{1}{N} \sum_{i=1}^{N} \sum_{u,v} g_i(u, v) \left\| [u, v]^\top - [u_i, v_i]^\top \right\|^2$$

This term encourages the distributions to be peaky, which has the added benefit of helping keep their means within the silhouette in the case of non-convex object boundaries.

3.4. 3D keypoint prediction network

There are many ways to design a neural network that can output a set of 3D keypoints given an input image. For example, one could employ a standard CNN with a fully-connected output layer to regress to an $N \times 3$ dimensional output. However, one important requirement for the mapping from images to keypoints is translation *equivariance* at the pixel level. That is, if we shift the input image, *e.g.,* to the left by one pixel, the output locations of all keypoints should also be changed by one unit. Training a standard CNN without this property would require a larger training set that contains objects at every possible location, while still providing no equivariance guarantees at the test time.

We propose the following simple modifications to achieve equivariance. Instead of regressing directly to the coordinate values, we ask the network to output a probability distribution map $g_i(u, v)$ that represents how likely keypoint i is to occur at pixel (u, v), with $\sum_{u,v} g_i(u, v) = 1$. We use a spatial softmax layer to produce a distribution over image pixels (Goroshin et al., 2015). We then compute the expected values of these spatial distributions to recover a pixel coordinate:

$$[u_i, v_i]^\top = \sum_{u,v} [u \cdot g_i(u, v), v \cdot g_i(u, v)]^\top \qquad (3)$$

For the z coordinates, we also ask the network to predict a depth value at every pixel, denoted $d_i(u, v)$, and compute

$$z_i = \sum_{u,v} d_i(u, v) g_i(u, v). \qquad (4)$$

We need to ensure that the resolutions of the input image and the probability map $g_i(u, v)$ are the same. Further, when the input image is translated by a fixed amount, the probability maps should also translate by the same amount. We achieve this by using a fully convolutional architectures (Long et al., 2015) also used for semantic segmentation. To increase the receptive field of the network, we stack multiple layers of *dilated* convolutions, similar to (Van Den Oord et al., 2016).

Our emphasis on designing an equivariance network not only helps significantly reduce the number of training examples required to achieve good generalization, but also takes the computational burden of converting between two representations (spatial-encoded in image to value-encoded in coordinates) off the network, so that it can focus on other critical tasks such as inferring depth.

Architecture details. Our architecture is structured as follows: The filter size for all of the layers is 3×3, and we stack 13 layers of dilated convolutions with dilation rates of $1, 1, 2, 4, 8, 16, 1, 2, 4, 8, 16, 1, 1$, all with 64 output channels except the last layer which has $2N$ output channels, split between g_i and d_i. We use leakyRelu and Batch Normalization (Ioffe & Szegedy, 2015) for all layers except the last layer. The output layers for d_i have no activation function, and the cannels are passed through a spatial softmax to produce g_i. Finally, g_i and d_i are then converted to actual coordinates p_i using Equations (3) and (4).

Breaking symmetry. Many object classes are symmetric across at least one axis, *e.g.,* the left side of a sedan looks like the right side flipped. This presents a challenge to the network because different parts can appear visually identical, and can only be resolved by understanding global context. For example, distinguishing the left wheels from the right wheels of a car when seeing from a side requires knowing its orientation (*i.e.,* whether the car is facing left or right). Both supervised and unsupervised techniques benefit from some global conditioning to aid in breaking ties and to make the task of keypoint prediction more deterministic.

To help break object symmetries, one can condition on some coarse quantization of the pose, and then predict the keypoint locations, based on which we can refine the pose further. Such a coarse to fine approach to detect keypoints is discussed in more depth in (Tulsiani & Malik, 2015). One simple conditioning signal that we use is a global binary flag that indicates whether the dominant direction of an object is facing left or right. This dominant direction, usually the $+x$ axis, comes directly from the dataset that we use (see Section 4), in which the 3D models are consistently oriented. To infer keypoints without this additional flag at test time, we train a network with the same architecture, although half the size, to predict this binary flag.

In particular, we train this network to predict the projected pixel locations of two 3D points $[1, 0, 0]$ and $[-1, 0, 0]$, transformed into each image's view in a training pair. These points correspond to the front and back of a normalized object. This network has a single L_2 loss between the predicted pixel locations and the ground-truth locations. The output binary flag is 1 if and only if the x coordinate of the projected pixel of the first point is greater than that of the second point. This flag is then fed into the keypoint prediction network.

4. Experiments

This section describes our evaluations, comparisons against related work, and results on various object classes.

Training data. Our training data is generated from the ShapeNet dataset (Chang et al., 2015), a large-scale database of 3D models with more than 51K models across 270 object categories. Each training set contains objects from a single category, *e.g.,* car, chair, and plane. For each model in each category, we generate 100 training pairs by rendering 200 images of size 128x128 under different viewing angles. The camera angles are first randomly sampled around the object from a fixed distance, all above the ground with zero roll angle. We then add small random shifts to the camera positions. All objects are normalized so that the longest dimension spans the range $[-1, 1]$. We use textures and materials defined in each mesh file and render with one hemisphere light directly above the object using Blender.

Implementation details. We implemented our network in TensorFlow (Abadi et al., 2015), and trained with the Adam optimizer with a learning rate of $10^{-3}, \beta_1 = 0.9, \beta_2 = 0.999$, and a total batch size of 256. We use the following weights for the losses: $(\alpha_{\text{con}}, \alpha_{\text{pose}}, \alpha_{\text{sep}}, \alpha_{\text{obj}}) = (1, 0.2, 1.0, 1.0)$. We train the network for $200K$ steps using synchronous training with 32 replicas.

4.1. Comparison with the supervised approach

To evaluate how well our unsupervised technique performs compared to a supervised technique, we created a database of human-label 3D landmarks for ShapeNet's cars using Amazon Mechanical Turk. For each car, we ask 3 different users to click on 12 points that most correspond to 12 reference points shown as an example to the users. These 12 reference points are based on the reference points used in the Pascal 3D+ dataset. We render the object from multiple views so that every point we ask the user to specify is facing outward from the screen. We then compute the average pixel locations for each keypoints, and ray cast back to the object to obtain the 3D coordinates of the keypoints.

Using these supervised labels, we train a network with the same architecture as in Section 3.4 to outputs keypoint locations in normalized coordinates $[-1, 1]$, as well as depths, using L_2 loss to the human labels. We then compute the angular distance error on a test set containing 720 cars with 72,000 pairs of ground-truth relative rotations. In Figure 2, we plot the histograms of angular errors of our method vs. the supervised technique, and show the error statistics in Table 1. For a fair comparison against the supervised technique, we provide an additional orientation flag to the supervised network. This is done by training another version of the supervised network that receives the orientation flag predicted from a pre-trained orientation network. Addi-

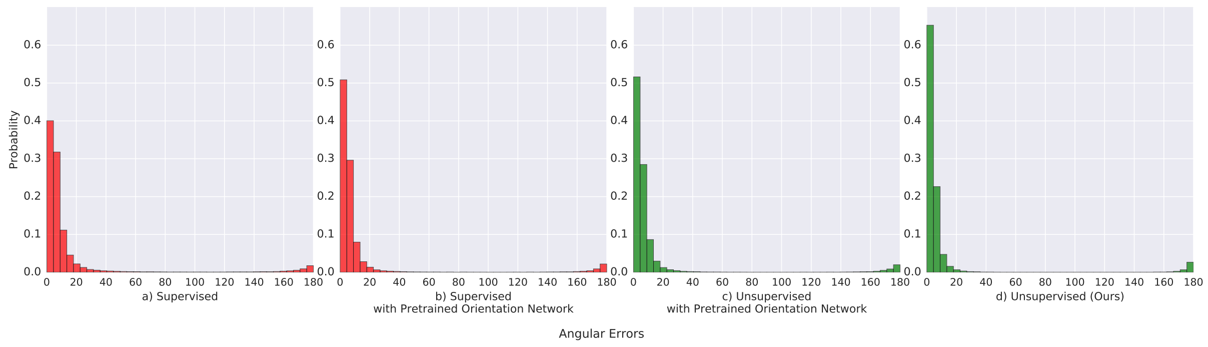

Figure 2. Histogram plots of angular distance errors between the ground-truth relative rotations and the least-squares estimates computed from two sets of keypoints predicted from test pairs. a) is a supervised method trained with a single L_2 loss between the pixel location prediction to the human labels. b) is the same as a) except the network is given an additional orientation flag predicted from a pretrained orientation network. c) is our network that uses the same pretrained orientation network as b), and d) is our unsupervised method trained jointly (the orientation and keypoint networks).

Method:	Mean	Median	3D-SE
a) Supervised	16.268	5.583	0.240
b) Supervised with Pretrained O-Net	13.961	4.475	0.197
c) Ours with Pretrained O-Net	13.500	4.418	0.165
d) **Ours**	11.310	3.372	0.171

Table 1. Mean and median angular distance errors between the ground-truth relative rotation and the Procrustes estimate computed from two sets of keypoints predicted on test image pairs. The standard errors (3D-SE) of predicted keypoints is described in Section 4.1. O-Net is the model that predicts a binary orientation.

tionally, we tested a more comparable version of our unsupervised network where we use and fix the same pre-trained orientation network during training.

Our unsupervised technique produces lower mean and median rotation errors than both versions of the supervised technique. Note that our technique sometimes incorrectly predicts keypoints that are 180° flipped from the right location due to the incorrect orientation prediction.

Keypoint location consistency. To evaluate the consistency of predicted keypoints across views, we transform the keypoints predicted for the same object under different views to the object space using the known camera matrices used for rendering. Then we compute the standard error of 3D locations for all keypoints across all test cars (Table 1). To disregard prediction outliers when the network incorrectly infers the orientation, we compute this metric only for keypoints whose rotation estimate error is less than 90° as an approximation (left halves of the histograms in Figure 2), for both the supervised and our unsupervised techniques.

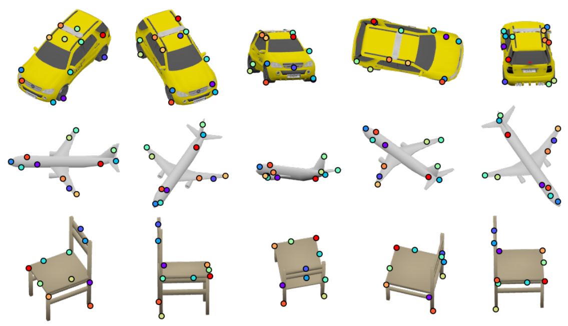

Figure 3. Keypoint results on single objects from different views. Note that these keypoints are predicted consistently across views even when they are completely occluded. (*e.g.,* the yellow point that tracks the back right leg of the chair.)

4.2. Generalization across views and instances

In this section, we show qualitative results of our keypoint predictions on cars, chairs, and planes. We trained our algorithm separately on 3 different object classes, and then tested on test models (10% held out from the collection). In Figure 3, we show keypoint prediction results on single objects from different views. Some of these views are quite challenging such as the top-down view of the chair. However, our network is able to infer the orientation and predict occluded parts such as the chair legs. In Figure 4, we run our network on many instances of test objects. Note that during training, the network only sees a pair of images of the same model, but it is able to utilize the same keypoints across semantically-similar parts across all instances from the same class. For example, the blue keypoints always track the cockpit of the planes.

Failure cases. When our orientation network fails to predict the correct orientation, the output keypoints will be

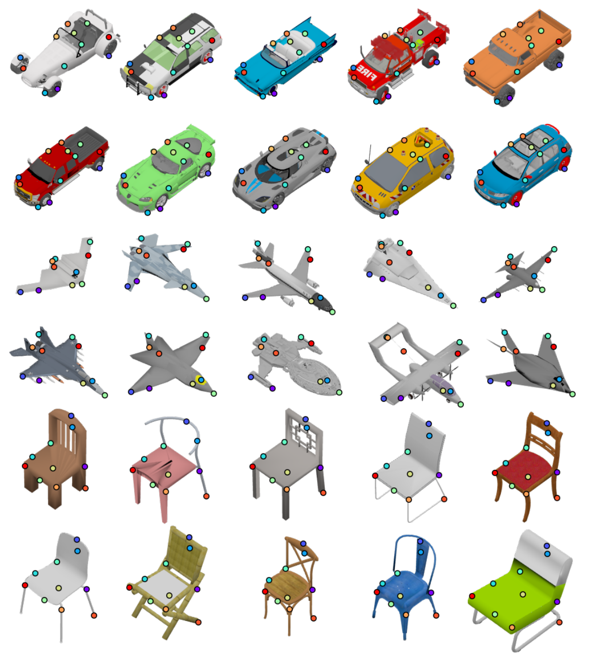

Figure 4. Results on ShapeNet (Chang et al., 2015) test set for cars, planes, and chairs. Our network is able to generalize across unseen appearances and shape variations, and consistently predict occluded parts such as wheels and chair legs. (yellow points). (Mo: we may want to make it clear that colored keypoints across different categories don't correspond to each other.)

flipped as shown in Figure 5 a). This happens for cars whose front and back look very similar, or for unusual wing shapes that make inference of the dominant direction hard.

Result on real images. We show a proof-of-concept inference result on real images from the Pascal3D+ (Xiang et al., 2014) dataset (Figure 5), which suggests that our proposed technique can be applicable across domains.

4.3. Ablation study

We present an ablation study for primary losses as well as how their weights affect the results. (Figure 6)

Removing multi-view consistency loss. This causes some of the keypoints to move around when the viewing angle changes, and not track onto any particular part of the object. The pose estimation loss alone may only provide a strong gradient for a number of keypoints as long as they give a good rotation estimate, but it does not explicitly force every point to be consistent.

Pose estimation loss & Noise. Removing pose estimation loss completely leads the network to place keypoints near the center of an object, which is the area with the least rotation motion, and thus least pixel displacement under different views. Increasing the noise that is added to the keypoints for rotation estimation encourages the keypoints to be spread apart from the center.

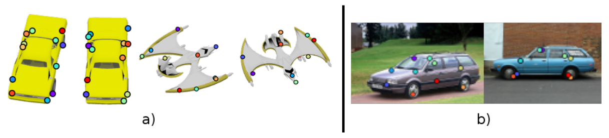

Figure 5. a) Failure cases when the orientation network predicts the wrong orientation. b) Proof-of-concept, hand-picked results on real images when trained with random backgrounds.

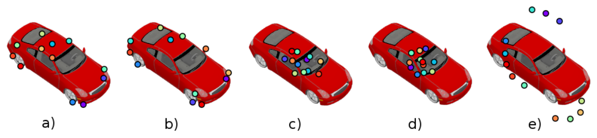

Figure 6. An ablation study for the losses. a) Our baseline model. b) and c) use twice the noise (0.2) and no noise respectively in the pose estimation loss. d) removes the pose estimation loss. e) removes Silhouette loss.

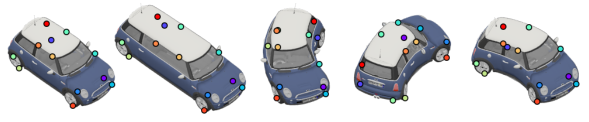

Figure 7. Results on a non-rigidly deformed car.

Removing silhouette consistency. This causes the keypoints to lie outside the object. Interestingly, the keypoints still satisfy multi-view consistency, and lie on a virtual 3D space that rotates with the object.

Result on deformed object. To evaluate the robustness of these keypoints under shape variations such as the length of the car, and whether the network uses local features to detect local parts as opposed to placing keypoints on a regular rigid structure, we run our network on a non-rigidly deformed car in Figure 7. Here we show that the network is able to predict where the wheels are and the overall deformation of the car structure.

5. Conclusion

We examine whether it is possible to optimize a representation based on a *sparse* set of keypoints or landmarks, without access to keypoint annotations, solely based on an end-to-end geometric reasoning framework. We show that indeed, one can discover semantically meaningful keypoints that are consistent across multiple views by formulating an objective function based on a 3D consistency loss and a relative pose estimation loss. Our translation equivariant architecture is able to generalize to unseen object instances of object classes from the synthetic ShapeNet (Chang et al., 2015) database. Importantly, our discovered keypoints outperform those from a direct supervised learning baseline on the problem of rigid 3D pose estimation.

References

Abadi, Martín, Agarwal, Ashish, Barham, Paul, Brevdo, Eugene, Chen, Zhifeng, Citro, Craig, Corrado, Greg S., Davis, Andy, Dean, Jeffrey, Devin, Matthieu, Ghemawat, Sanjay, Goodfellow, Ian, Harp, Andrew, Irving, Geoffrey, Isard, Michael, Jia, Yangqing, Jozefowicz, Rafal, Kaiser, Lukasz, Kudlur, Manjunath, Levenberg, Josh, Mané, Dan, Monga, Rajat, Moore, Sherry, Murray, Derek, Olah, Chris, Schuster, Mike, Shlens, Jonathon, Steiner, Benoit, Sutskever, Ilya, Talwar, Kunal, Tucker, Paul, Vanhoucke, Vincent, Vasudevan, Vijay, Viégas, Fernanda, Vinyals, Oriol, Warden, Pete, Wattenberg, Martin, Wicke, Martin, Yu, Yuan, and Zheng, Xiaoqiang. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`. Software available from tensorflow.org.

Agrawal, Pulkit, Carreira, Joao, and Malik, Jitendra. Learning to see by moving. In *ICCV*, 2015.

Andriluka, Mykhaylo, Pishchulin, Leonid, Gehler, Peter, and Schiele, Bernt. 2D human pose estimation: New benchmark and state of the art analysis. In *CVPR*, June 2014.

Arie-Nachimson, M. and Basri, R. Constructing implicit 3d shape models for pose estimation. In *ICCV*, 2009.

Chang, Angel X., Funkhouser, Thomas, Guibas, Leonidas, Hanrahan, Pat, Huang, Qixing, Li, Zimo, Savarese, Silvio, Savva, Manolis, Song, Shuran, Su, Hao, Xiao, Jianxiong, Yi, Li, and Yu, Fisher. ShapeNet: An Information-Rich 3D Model Repository. *arXiv:1512.03012*, 2015.

Chen, Ching-Hang and Ramanan, Deva. 3D human pose estimation= 2D pose estimation+ matching. In *CVPR*, 2017.

Chen, Yu, Shen, Chunhua, Wei, Xiu-Shen, Liu, Lingqiao, and Yang, Jian. Adversarial learning of structure-aware fully convolutional networks for landmark localization. *arXiv:1711.00253*, 2017.

Choy, Christopher B, Gwak, JunYoung, Savarese, Silvio, and Chandraker, Manmohan. Universal correspondence network. In *NIPS*, 2016.

Dalal, Navneet and Triggs, Bill. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

Goroshin, Ross, Mathieu, Michael F, and LeCun, Yann. Learning to linearize under uncertainty. In *NIPS*, pp. 1234–1242, 2015.

Güler, Rıza Alp, Neverova, Natalia, and Kokkinos, Iasonas. Densepose: Dense human pose estimation in the wild. *arXiv:1802.00434*, 2018.

Han, Kai, Rezende, Rafael S, Ham, Bumsub, Wong, Kwan-Yee K, Cho, Minsu, Schmid, Cordelia, and Ponce, Jean. SCNet: Learning semantic correspondence. In *ICCV*, 2017.

He, Kaiming, Zhang, Xiangyu, Ren, Shaoqing, and Sun, Jian. Deep residual learning for image recognition. *CVPR*, 2016.

He, Kaiming, Gkioxari, Georgia, Dollár, Piotr, and Girshick, Ross. Mask R-CNN. In *ICCV*, 2017.

Hejrati, Mohsen and Ramanan, Deva. Analyzing 3d objects in cluttered images. In Pereira, F., Burges, C. J. C., Bottou, L., and Weinberger, K. Q. (eds.), *Advances in Neural Information Processing Systems 25*, pp. 593–601. Curran Associates, Inc., 2012.

Hinton, Geoffrey E, Krizhevsky, Alex, and Wang, Sida D. Transforming auto-encoders. In *Int. Conf. on Artificial Neural Networks*. Springer, 2011.

Huang, Qi-Xing and Guibas, Leonidas. Consistent shape maps via semidefinite programming. In *Computer Graphics Forum*, 2013.

Huang, Shaoli, Gong, Mingming, and Tao, Dacheng. A coarse-fine network for keypoint localization. In *ICCV*, 2017.

Ioffe, Sergey and Szegedy, Christian. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *ICML*, 2015.

Krizhevsky, Alex, Sutskever, Ilya, and Hinton, Geoffrey E. ImageNet classification with deep convolutional neural networks. *NIPS*, 2012.

LeCun, Yann, Bottou, Léon, Bengio, Yoshua, and Haffner, Patrick. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998.

Lepetit, Vincent, Moreno-Noguer, Francesc, and Fua, Pascal. EPnP: An accurate O(n) solution to the PnP problem. *IJCV*, 2008.

Lin, Tsung-Yi, Maire, Michael, Belongie, Serge, Hays, James, Perona, Pietro, Ramanan, Deva, Dollár, Piotr, and Zitnick, C Lawrence. Microsoft COCO: Common objects in context. In *ECCV*, 2014.

Long, Jonathan, Shelhamer, Evan, and Darrell, Trevor. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015.

Lowe, David G. Object recognition from local scale-invariant features. In *CVPR*, 1999.

Martinez, Julieta, Hossain, Rayat, Romero, Javier, and Little, James J. A simple yet effective baseline for 3D human pose estimation. In *ICCV*, 2017.

Mehta, Dushyant, Rhodin, Helge, Casas, Dan, Fua, Pascal, Sotnychenko, Oleksandr, Xu, Weipeng, and Theobalt, Christian. Monocular 3D human pose estimation in the wild using improved CNN supervision. In *3DV*, 2017a.

Mehta, Dushyant, Sotnychenko, Oleksandr, Mueller, Franziska, Xu, Weipeng, Sridhar, Srinath, Pons-Moll, Gerard, and Theobalt, Christian. Single-shot multi-person 3D body pose estimation from monocular RGB input. *arXiv:1712.03453*, 2017b.

Mehta, Dushyant, Sridhar, Srinath, Sotnychenko, Oleksandr, Rhodin, Helge, Shafiei, Mohammad, Seidel, Hans-Peter, Xu, Weipeng, Casas, Dan, and Theobalt, Christian. VNect: Real-time 3D Human Pose Estimation with a Single RGB Camera. volume 36, July 2017c.

Newell, Alejandro, Yang, Kaiyu, and Deng, Jia. Stacked hourglass networks for human pose estimation. In *ECCV*, 2016.

Papandreou, George, Zhu, Tyler, Kanazawa, Nori, Toshev, Alexander, Tompson, Jonathan, Bregler, Chris, and Murphy, Kevin. Towards accurate multiperson pose estimation in the wild. *arXiv:1701.01779*, 8, 2017.

Pishchulin, Leonid, Insafutdinov, Eldar, Tang, Siyu, Andres, Bjoern, Andriluka, Mykhaylo, Gehler, Peter, and Schiele, Bernt. DeepCut: Joint subset partition and labeling for multi person pose estimation. In *CVPR*, June 2016.

Ramakrishna, Varun, Kanade, Takeo, and Sheikh, Yaser. Reconstructing 3d human pose from 2d image landmarks. In *ECCV*. Springer, 2012.

Sabour, Sara, Frosst, Nicholas, and Hinton, Geoffrey E. Dynamic routing between capsules. In *NIPS*, 2017.

Sagonas, Christos, Antonakos, Epameinondas, Tzimiropoulos, Georgios, Zafeiriou, Stefanos, and Pantic, Maja. 300 faces in-the-wild challenge: Database and results. *Image and Vision Computing*, 47, 2016.

Salti, Samuele, Tombari, Federico, Spezialetti, Riccardo, and Di Stefano, Luigi. Learning a descriptor-specific 3d keypoint detector. In *ICCV*, 2015.

Schönemann, Peter. A generalized solution of the orthogonal Procrustes problem. *Psychometrika*, 31(1), 1966.

Snavely, Noah, Seitz, Steven M, and Szeliski, Richard. Photo tourism: exploring photo collections in 3d. In *ACM transactions on graphics (TOG)*, 2006.

Su, Hao, Qi, Charles R, Li, Yangyan, and Guibas, Leonidas J. Render for CNN: Viewpoint estimation in images using CNNs trained with rendered 3D model views. In *ICCV*, 2015.

Thewlis, James, Bilen, Hakan, and Vedaldi, Andrea. Unsupervised learning of object frames by dense equivariant image labelling. In *NIPS*, 2017.

Tompson, Jonathan, Stein, Murphy, Lecun, Yann, and Perlin, Ken. Real-time continuous pose recovery of human hands using convolutional networks, journal = ACM Transactions on Graphics, year = 2014, month = August, volume = 33.

Tompson, Jonathan J, Jain, Arjun, LeCun, Yann, and Bregler, Christoph. Joint training of a convolutional network and a graphical model for human pose estimation. In *NIPS*, 2014.

Toshev, Alexander and Szegedy, Christian. Deeppose: Human pose estimation via deep neural networks. In *CVPR*, 2014.

Tulsiani, Shubham and Malik, Jitendra. Viewpoints and keypoints. In *CVPR*, 2015.

Van Den Oord, Aaron, Dieleman, Sander, Zen, Heiga, Simonyan, Karen, Vinyals, Oriol, Graves, Alex, Kalchbrenner, Nal, Senior, Andrew, and Kavukcuoglu, Koray. Wavenet: A generative model for raw audio. *arXiv:1609.03499*, 2016.

Wu, Jiajun, Xue, Tianfan, Lim, Joseph J, Tian, Yuandong, Tenenbaum, Joshua B, Torralba, Antonio, and Freeman, William T. Single Image 3D Interpreter Network. In *ECCV*, 2016.

Xiang, Yu, Mottaghi, Roozbeh, and Savarese, Silvio. Beyond PASCAL: A Benchmark for 3D Object Detection in the Wild. In *WACV*, 2014.

Yang, Wei, Li, Shuang, Ouyang, Wanli, Li, Hongsheng, and Wang, Xiaogang. Learning feature pyramids for human pose estimation. In *ICCV*, volume 2, 2017.

Zhou, Tinghui, Krahenbuhl, Philipp, Aubry, Mathieu, Huang, Qixing, and Efros, Alexei A. Learning dense correspondence via 3d-guided cycle consistency. In *CVPR*, 2016a.

Zhou, Xiaowei, Zhu, Menglong, and Daniilidis, Kostas. Multi-image matching via fast alternating minimization. In *CVPR*, 2015.

Zhou, Xiaowei, Zhu, Menglong, Leonardos, Spyridon, Derpanis, Konstantinos G, and Daniilidis, Kostas. Sparseness meets deepness: 3D human pose estimation from monocular video. In *CVPR*, 2016b.

Zhou, Xingyi, Karpur, Arjun, Gan, Chuang, Luo, Linjie, and Huang, Qixing. Unsupervised domain adaptation for 3d keypoint prediction from a single depth scan. *arXiv:1712.05765*, 2017.