Abstract We consider the task of estimating the 3D orientation of an object of known category given an image of the object and a bounding box around it. Recently, CNN-based regression and classification methods have shown significant performance improvements for this task. This paper proposes a new CNN-based approach to monocular orientation estimation that advances the state of the art in four different directions. First, we take into account the Riemannian structure of the orientation space when designing regression losses and nonlinear activation functions. Second, we propose a mixed Riemannian regression and classification framework that better handles the challenging case of nearly symmetric objects. Third, we propose a data augmentation strategy that is specifically designed to capture changes in 3D orientation. Fourth, our approach leads to state-of-the-art results on the PASCAL3D+ dataset.

1 Introduction

A long-standing goal of computer vision is to teach a machine to “see” and understand the 3D world captured by a 2D image. One way to do this is to describe the image in terms of the objects present in it and recover the underlying geometry of the scene. This involves predicting the rigid transformations between the camera and objects in the image. This problem, known as 3D pose estimation, is an integral part of many problems in computer vision, e.g., scene understanding and reconstruction, and has recently seen renewed interest due to its applications in autonomous driving, robot manipulation and augmented reality, where the ability to reason and plan in 3D is of vital importance.

The “3D pose” of an object consists of its 3D location (described by an extrinsic translation $T$) and its 3D orientation (described by an extrinsic rotation $R$). In this work, we are interested only in estimating the 3D orientation of the object. Specifically, given an image and a bounding box around an object in the image with known object category label, we consider the problem of estimating the object’s orientation (see Fig. 1). This simplification is motivated by the remarkable success of current object detection algorithms at predicting the object’s scale/depth and its 2D location. Therefore, we will assume that the bounding box and category label are either given by an oracle (ground-truth) or are the output of a detection system. Throughout this paper we use the words 3D pose and orientation interchangeably.

1.1 Prior Work

As we will discuss in more detail in §2, state-of-the-art methods for monocular object pose estimation are all based on convolutional neural networks (CNNs). One family of methods uses a CNN to extract keypoints related to the object in the image and then uses a PnP algorithm to predict the object’s pose from these keypoints. Another family of methods uses a CNN to predict the pose directly from the image. Such methods follow either a regression framework, where orientation is predicted directly, or a classification framework, where the orientation space is first discretized. Our work falls
Fig. 1: Overview of the problem statement: Given a 2D image $x_n$ of an object and a bounding box containing the object, design a CNN that predicts the object’s 3D pose $y_n$. As part of the problem, we try to answer the following questions with geometrically meaningful components for the CNN pipeline: (i) What is an appropriate representation of 3D pose? (ii) What is a good loss function for this task? (iii) What kind of network architectures are useful for this task?? (iv) What data augmentation strategies are useful for this task?

in the second family of methods as we are also interested in predicting the pose directly from the input image.

In our view, a major disadvantage of existing methods is that they use representations of orientation and loss functions that do not properly take into account the geometry of the space of rotations. For example, many orientation estimation methods represent orientation in terms of Euler angles, discretize the angles into bins and solve a pose classification task while training the network with a cross-entropy loss. While some methods try to incorporate the cyclical symmetry of angles via a weighted cross-entropy loss, this is very different from using the Riemannian geometry of the orientation space. Other methods use better representations, but they still discretize the orientation space to eventually solve a classification problem. The drawback of classification-based approaches is that they give non-zero estimation error even with perfect pose classification accuracy due to the discretization process. Moreover, this error might be large if the binning is coarse.

Regression-based approaches overcome errors due to binning by predicting a continuous pose. However, current methods either regress a $(\cos, \sin)$ representation of Euler angles or use a Euclidean distance between orientation representations, which again ignores Riemannian properties like geodesic distances on the space of orientations. Another issue with regression-based methods is that they predict a single orientation without a confidence score. This is a problem for object categories that exhibit strong symmetries because the regression system will predict a single orientation which might be far from the ground truth. In such cases, it would be preferable to have a system that predicts multimodal orientation distributions with a confidence score for each hypothesis. Finally, another disadvantage of existing methods is that network architectures and training methods are not specifically designed for orientation estimation. For example, existing data augmentation techniques such as 2D jittering are adequate for object classification and detection, but some form of 3D jittering is needed for orientation estimation.

1.2 Paper Contributions

This work addresses the aforementioned drawbacks by designing a new 3D orientation estimation system that incorporates the geometry of the space of rotations into a combined regression and classification framework.

Our first contribution is the use of orientation representations like axis-angle and quaternion as outputs of a CNN and geodesic distances on the space of rotations as loss functions of our CNN. While these representations and distances have been used extensively in computer vision, see e.g., Ma et al (2003), it is only recently that they have been used in the context of CNNs, e.g., Kendall et al (2015) use the quaternion representation with an Euclidean loss function.

Our second contribution is a framework for combining regression and classification-based approaches to orientation estimation. As we discussed earlier, pure regression and classification-based approaches both have certain disadvantages. However, by combining both formulations, we can get the best of both worlds. Specifically, we propose a family of Bin & Delta models, where a Bin network predicts a discrete orientation by solving a classification task and a Delta network predicts a continuous refinement of the discrete orientation by solving a regression task. The proposed framework is flexible and allows for many choices of regression and/or classification losses, what the output of each network represents and how to combine them. While recent work does combine pose regression and classification methods (see §2 for details), such methods are particular cases of our more general framework, which is designed taking the geometry of the space of rotations into account.

Our third contribution is a 3D pose jittering strategy for the task of orientation estimation. 2D jittering involves translating the bounding box around an object
in the image plane, which is adequate for 2D object detection and classification, but limited for 3D orientation estimation. In this work, we jitter the object in 3D using small rotations (both in-plane rotations via deviations of camera-tilt and out-of-plane rotations via deviations of azimuth and elevation angles). This is more appropriate to train orientation estimation networks and better reflects the geometry of the jittered augmented data.

Our fourth contribution is to obtain state-of-the-art performance on the challenging Pascal3D+ dataset under a variety of metrics using both ground-truth bounding boxes and bounding boxes returned by object detection systems. Specifically, we introduce 14 distinct Bin & Delta models with different decision choices and compare their performance. We also present a detailed evaluation of various decision choices via an extensive ablation analysis. This is an empirical validation of our design choices and the advantages of using a geometrically-aware CNN system for orientation estimation.

To summarize, the contributions of our work are:

- The integration of Riemannian representations of rotations and geodesic loss functions for 3D orientation estimation into convolutional neural networks;
- The integration of regression and classification approaches to predict continuous 3D orientations while modeling multimodal orientation distributions;
- A new geometry-based data augmentation strategy that is appropriate for orientation estimation; and
- An experimental evaluation of our models showing that they achieve state-of-the-art performance on the challenging Pascal3D+ dataset.

This paper is an extended version of our preliminary work (Mahendran et al., 2017, 2018) with new models and architectures, new activation and loss functions, and new experimental results. Specifically, we introduce the Riemannian Bin & Delta models in §5.3.2 that improve upon the Log-Euclidean Bin & Delta models in Mahendran et al. (2018) (§5.3.7 here). We also recognize two different relaxations in the Probabilistic Bin & Delta models in Mahendran et al. (2018) and correspondingly split them into our Probabilistic Bin & Delta (§5.3.3), ReLaXed Bin & Delta (§5.3.4) and ReLaXed Probabilistic Bin & Delta models (§5.3.5). We also expand our ablation analysis (§7.4) relative to Mahendran et al. (2017) with new results from our Geodesic Bin & Delta models. Finally, we also expand results on orientation estimation with detected bounding boxes (§7.5) relative to Mahendran et al. (2017) with new models and a more detailed evaluation with different object detection systems.

1.3 Paper Organization

The remainder of the paper is organized as follows. In §2, we review the state of the art on 3D pose estimation. In §3, we review some geometric properties of the space of orientations that we would like to use in designing our networks for orientation estimation, namely, representations of orientation in §3.1 and geodesic loss functions in §3.2. Then in §4, we describe our proposed geodesic regression network for orientation estimation that uses these representations and loss functions. In §5, we discuss our proposed Bin & Delta models (outlined in Table 1) that combine classification and regression for orientation estimation. This is followed by §6 where we explain our geometry-based data augmentation strategy. In §7, we demonstrate the effectiveness of our models with state-of-the-art performance on the Pascal3D+ dataset and a detailed experimental evaluation that includes results with bounding boxes returned by an oracle §7.3 and an object detection system §7.5. Finally, in §8, we state conclusions derived from our work.

2 Related Work

3D Pose Estimation Algorithms. There are many non-deep learning methods for 3D pose estimation given 2D images (Lpezsastre et al., 2011; Hejrati and Ramanan, 2012, 2014; Aubry et al., 2014; GLasner et al., 2011; Liebelt and Schmid, 2010; Lim et al., 2013, 2014; Pepik et al., 2012b,a; Savarese and Fei-Fei, 2007, 2008). However, due to space constraints, we restrict our review to methods based only on deep networks. As mentioned earlier, the current literature on 3D pose estimation using deep networks can be divided in two groups: (i) methods that predict 2D keypoints from images and then recover 3D pose by solving a PnP problem, and (ii) methods that directly predict 3D pose from an image.

The first group of methods includes the works of Crivellaro et al. (2015); Grabner et al. (2018); Pavlakos et al. (2017); Wu et al. (2016); Rad and Lepetit (2017). Pavlakos et al. (2017) and Wu et al. (2016) use 2D keypoints corresponding to the projections of semantically meaningful points on 3D object models as the output of their network. Given a new image, they predict a probabilistic map of 2D keypoints and recover 3D pose by comparing with some pre-defined object models. In Grabner et al. (2018); Rad and Lepetit (2017); Crivellaro et al. (2015), instead of semantic keypoints, they use 2D keypoints corresponding to the projection of the 8 corners of a 3D bounding box encapsulating the object. The network is trained by comparing the predicted
2D keypoint locations with the projections of the 3D keypoints on the image under ground-truth pose annotations. Grabner et al (2018) uses a Huber loss on the projection error to be robust to inaccurate ground-truth annotations and is the current state-of-the-art on the Pascal3D+ dataset to the best of our knowledge.

The second group of methods includes the works of Tulsiani and Malik (2015); Su et al (2015); Elhosseiny et al (2016); Massa et al (2016, 2014); Wang et al (2016); Mousavian et al (2017). All these methods train a network to predict the orientation given 2D images and use the Euler angle representation of rotation matrices to estimate the azimuth, elevation and camera-tilt angles separately. Tulsiani and Malik (2015) and Elhosseiny et al (2016) divide the angles into non-overlapping bins and solve a classification problem. Su et al (2015) additionally use a weighted cross-entropy loss for fine-grained orientation classification. On the other hand, Wang et al (2016) regresses the angles directly with a direct Euclidean loss. Massa et al (2016) proposes multiple loss functions based on regression and classification for joint object detection and pose estimation, and concludes that the classification methods work better than the regression ones. Mousavian et al (2017) divides the angles into overlapping bins and trains their network using a multibin loss that combines regression and classification. As we discussed earlier, our work is closest to the second group of methods in that we are also interested in a monocular orientation estimation problem, but with the key difference that we are interested in designing our orientation estimation system with representations, loss functions and data augmentation that is geometrically coherent.

**3D Pose Representations and Loss Functions.** There has been recent interest in quaternion representations of 3D pose for camera localization (Kendall et al, 2015; Kendall and Cipolla, 2016, 2017; Tulsiani et al, 2018), but these works recommend a Euclidean, re-projection or classification loss respectively instead of the geodesic loss. Another work (Wang et al, 2018), uses quaternion representation (with a smoothed $L_1$ loss) in combination with a 3D location field to predict 3D pose of cars within a joint object detection and pose estimation framework. Recently, Hou et al (2018) also propose Riemannian loss and gradients on the $SE(3)$ manifold in the context of CNNs for the task of pose estimation for image registration of fetal MRIs.

Our proposed mixed classification-regression framework, with its Bin and Delta models, can be considered a generalization of (Mousavian et al, 2017; Li et al, 2018; Güler et al, 2017, 2018). Specifically, Mousavian et al (2017) is a variation of the Geodesic Bin and Delta model we propose in Eqn. (16) with a cos–sin representation of Euler angles, while Li et al (2018) is a particular case of the Simple Bin and Delta model we propose in Eqn. (33) with a quaternion representation of 3D pose. On the other hand, the quantized regression model of Güler et al (2017, 2018) uses the Bin and Delta model to generate dense correspondences between a 3D model and an image for face landmark and human pose estimation. Güler et al (2017) learns a modification of our Simple Bin and Delta model in Eqn. (32) with a separate Delta network for every facial region while Güler et al (2018) is a particular case of the Probabilistic Bin and Delta model we propose in Eqn. (25). Güler et al (2018) also makes a connection between a Bin and Delta model and a mixture of regression experts proposed in Jordan and Jacobs (1994), where the classification output probability vector acts as a gating function on regression experts. There has been recent interest

| Name | Models | Section | Salient Features |
|------|--------|---------|-----------------|
| Geodesic Regression | RG | §4.3 | Geodesic regression loss on 3D pose outputs |
| Euclidean Regression | RE | §4.4.1 | Euclidean regression loss on 3D pose outputs |
| Classification | C | §4.4.2 | Classification on the discretized 3D pose space |
| Geodesic Bin & Delta | MG & MG+ | §5.3.1 | Bin & Delta model with geodesic loss on the 3D pose outputs and cross-entropy loss on the discretized pose space |
| Riemannian Bin & Delta | MR & MR+ | §5.3.2 | A Geodesic Bin & Delta model with a different (Riemannian) choice of Delta representation |
| Probabilistic Bin & Delta | MP & MP+ | §5.3.3 | A Geodesic Bin & Delta model with a probabilistic weighting of the geodesic loss |
| Relaxed Bin & Delta | MX & MX+ | §5.3.4 | A Geodesic Bin & Delta with a soft-assignment to pose labels and KL-divergence loss |
| Relaxed Probabilistic Bin & Delta | MXP & MXP+ | §5.3.5 | A Geodesic Bin & Delta model with both probabilistic weighting of geodesic loss and soft-assignment to pose-labels |
| Simple Bin & Delta | MS & MS+ | §5.3.6 | Bin & Delta model with Euclidean regression loss on the delta outputs and cross-entropy loss on the discretized pose space |
| Log-Euclidean Bin & Delta | MLE & MLE+ | §5.3.7 | Bin & Delta model with Log-Euclidean regression loss on the delta outputs and cross-entropy loss on the discretized pose space |

Table 1: An overview of our proposed models for monocular 3D pose estimation.
in trying to design networks and representations that combine classification and regression to model 3D pose but the authors of these different works have treated this as a one-off representation problem. In contrast, we propose a general framework that encapsulates prior models as particular cases.

3 Preliminaries on 3D Orientations

In this section, we provide a brief overview of representations and loss functions of orientations that respect the special geometry of the space of orientations. We direct the reader to Ma et al (2003) and Hartley and Zisserman (2004) for more details.

3.1 Representations of Orientation

We are interested in estimating the orientation, represented by a 3D rotation $R$, of an object in the image. There are multiple ways in which 3D rotations can be represented and we describe three of them here: (1) Euler angles, (2) Axis-angles and (3) Quaternions. We use the axis-angle representation in this work as it best captures the geometry and complexity of the space of 3D orientations.

**Euler Angles.** A common way of describing rotation matrices is in terms of its Euler angles: azimuth $az$, elevation $el$ and camera-tilt $ct$. Assuming a ZXZ convention, the rotation matrix $R$ is given by

$$
R = \begin{pmatrix}
cc - se & 0 & 0 \\
sc & cc & 0 \\
0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
ca - sa & 0 & 0 \\
sa & ca & 0 \\
0 & 0 & 1
\end{pmatrix},
$$

where $ca = \cos(az)$, $sa = \sin(az)$, $cc = \cos(el)$, $se = \sin(el)$, $cc = \cos(ct)$, $sc = \sin(ct)$ and the three matrices capture rotations about the $Z$-axis by angle $ct$, about the $X$-axis by angle $el$ and about the $Z$-axis by angle $az$ respectively. However, this representation of 3D rotations, $(az, el, ct)$, suffers from the following two drawbacks: (1) The representation is not unique when $el = 0$, a problem also commonly known as the Gimbal Lock. (2) The representation ignores properties of the space rotation matrices. Specifically, any rotation matrix $R$ lies in the set of special orthogonal matrices $SO(3) \doteq \{ R : R \in \mathbb{R}^{3 \times 3}, R^T R = I_3, \det(R) = 1 \}$. This space is a Lie group with an underlying Riemannian geometry, geodesic distance, etc. which are ignored in the per-angle representations and loss functions. Instead, we propose the use of two representations, axis-angle and quaternion, that better capture these properties.

**Axis-angle.** A rotation matrix $R$ captures the rotation of 3D points by an angle $\theta \in [-\pi, \pi]$ about a unit-norm axis $v \in \mathbb{R}^3$. This can be expressed as $R = \expm(\theta [v]_\times)$, where $\expm$ is the matrix exponential and $[v]_\times$ is the skew-symmetric operator of vector $v$, i.e. $[v]_\times = \begin{bmatrix} 0 & -v_3 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{bmatrix}$ for $v = [v_1, v_2, v_3]^T$ such that $||v||_2 = 1$. This can also be expressed as the exponential map $\exp : \mathbb{R}^3 \rightarrow SO(3)$ between axis-angle vectors $y \in \mathbb{R}^3$ and rotations $R \in SO(3)$. Similarly, we can also define a logarithm map between a rotation $R$ and its axis-angle vector $y = \theta v$, $\log : SO(3) \rightarrow \mathbb{R}^3$. Restrict $\theta \in [0, \pi]$ and defining $R = I_3 \Leftrightarrow y = 0_3$ ensures an invertible mapping.

**Quaternion.** Another popular representation for 3D rotation matrices are quaternions. Given an axis-angle vector $y = \theta v$, the corresponding quaternion $q = (c, s)$ is given by $(\cos \frac{\theta}{2}, \sin \frac{\theta}{2} v)^T$. By construction, quaternions are unit norm, i.e. $||q||_2 = 1$, and are points on the hypersphere $S^3 \doteq \{ x : x \in \mathbb{R}^4, ||x||_2 = 1 \}$. Also, note that quaternions $q$ and $-q$, or antipodal points on the hypersphere $S^3$, correspond to rotations by angle $\theta$ about axis $v$ and angle $2\pi - \theta$ about axis $-v$ respectively, which correspond to the same rotation matrix. Axis-angle vectors represent 3-degree of freedom rotations with a compact 3-dimensional representation whereas quaternions use a 4-dimensional representation with an additional unit-norm constraint.

3.2 Loss functions on 3D Orientation space

As we mentioned earlier, we would like to exploit the special properties of rotation matrices. One such property is the geodesic distance or the shortest distance between two points along the manifold, which is a better measure of the “closeness” between two elements of the manifold when compared to the Euclidean distance. The geodesic distance between two rotation matrices $R_1$ and $R_2$ is given by

$$
d(R_1, R_2) = \frac{||\logm(R_1 R_2^T)||_F}{\sqrt{2}},
$$

where $\logm$ is the matrix logarithm and $|| \cdot ||_F$ is the Frobenius norm. This can be simplified further using the Rodrigues’ rotation formula,

$$
R = I_3 + \sin \theta [v]_\times + (1 - \cos \theta) [v]_\times^2,
$$

to get

$$
d(R_1, R_2) = |\cos^{-1}\left[ \frac{\text{tr}(R_1^T R_2) - 1}{2} \right] |.
$$

This geodesic distance better captures the (Riemannian) geometry of the orientation space and is the loss function we would like to use for our proposed work.
Fig. 2: Overall network architecture which takes image \( x \) and object category label \( c \) as input and predicts object pose \( y \). The feature network \( \Phi_F(\cdot; W_F) \) is shared across all object categories, while each category has its own pose network \( \Phi_P(\cdot; W_P^c) \).

### 4 Geodesic Regression Network for Orientation Estimation

In this section, we present our proposed network that formulates the orientation estimation task as a geodesic regression problem. We present the network architecture in §4.1, loss function in §4.2 and model in §4.3. We also present two baselines that formulate the task as Euclidean regression (§4.4.1) or classification (§4.4.2).

#### 4.1 Architecture

We follow a standard network design for 3D pose estimation, where the network has two parts: a feature network and a pose network, as illustrated in Fig. 2. The feature network is shared across all object categories, but there is one pose network associated with each object category. The feature network takes as input an Image \( x \), and outputs the feature descriptor of the image, \( \Phi_F(x; W_F) \), where \( W_F \) are the weights associated with the feature network. Our feature network is the ResNet-50 network of He et al (2016a,b) minus the last classification layer. The pose network takes the object category label \( c \) and the output of the feature network \( f = \Phi_F(x; W_F) \) as input and predicts the object pose \( y = \Phi_P(f; W_P^c) \), where \( W_P^c \) denotes weights of the pose network associated with category label \( c \).

The pose network has 3 fully connected layers with associated ReLU activations and batch normalization as outlined in Fig. 3 for the axis-angle representation. The difference between different pose networks for axis-angle and quaternion representations is in the last 2 layers as shown in Fig. 4. For the axis-angle representation, the output of the pose network is a 3-dim output \( y = \theta v \) and we model the constraints \( \theta \in [0, \pi] \) and \( v_i \in [-1, 1] \) using a \( \pi \) tanh non-linearity. For the quaternion representation, the output is 4-dimensional and we model the unit-norm constraint using a \( L_2 \) normalization non-linearity.

#### 4.2 Loss Functions

We would like to use the geodesic regression loss discussed in §3.2 as our loss function for Geodesic Regression. Under the two representations of axis-angle and quaternion, this loss function takes different forms.

**Axis-angle Loss.** The geodesic regression loss between two axis-angle vectors \( y_1 \) and \( y_2 \) is given by

\[
\mathcal{L}_R(y_1, y_2) \equiv d(R_1, R_2),
\]

where \( R_1 = \exp(y_1) \) and \( R_2 = \exp(y_2) \) and \( d(R_1, R_2) \) is defined in Eqn. (4). This geodesic regression loss is our proposed alternative to the standard Euclidean regression loss, \( \mathcal{L}_E(y_1, y_2) = \|y_1 - y_2\|^2 \).

Note that another interpretation of axis-angle vectors \( y = \log(R) \) is that they are projections of rotation matrices \( R \) to the tangent space at the identity \( I_3 \). Under this interpretation of axis-angle vectors, the Euclidean regression loss \( \mathcal{L}_E(y_1, y_2) \) is (locally) the Euclidean distance between projections of corresponding rotation matrices \( R_1 \) and \( R_2 \) in the tangent space at \( I_3 \). The Euclidean distance on the tangent space is also a popular choice of distances in Riemannian geometry but, this is a good distance only locally (near \( I_3 \)) whereas the geodesic distance is valid globally.
It is interesting to note that,

\[ \mathcal{L}_R(y_1, y_2) = d(R_1, R_2) = \frac{\| \logm(R_1^T R_2) \|_F}{\sqrt{2}} \]

\[ = \frac{1}{\sqrt{2}} \| \logm [ \expm(\theta_1[v_1]) \expm(\theta_2[v_2])^T ] \|_F \]

\[ \neq \frac{1}{\sqrt{2}} \| \logm [ \expm(\theta_1[v_1] - \theta_2[v_2]) ] \|_F \]

\[ = \frac{1}{\sqrt{2}} \| \theta_1[v_1] - \theta_2[v_2] \|_F = \| \theta_1v_1 - \theta_2v_2 \|_2 \]

\[ = \| y_1 - y_2 \| = \mathcal{L}_E(y_1, y_2) \quad (6) \]

This is another way of saying that the Euclidean distance between two axis-angle vectors is not a geometrically appropriate loss function.

**Quaternion Loss.** The geodesic distance between two quaternions \( q_1 \) and \( q_2 \) is given by

\[ d(q_1, q_2) = 2 \cos^{-1}(|c|), \quad \text{where} \quad (c, s) = q_1^{-1} \cdot q_2. \quad (7) \]

This is identical to the geodesic distance \( d(R_1, R_2) \) of Eqn.\,(2), where \( q_1 \) and \( q_2 \) are quaternions corresponding to rotation matrices \( R_1 \) and \( R_2 \) respectively. Using quaternion algebra, we have \((c_1, s_1) \cdot (c_2, s_2) = (c_1c_2 - \langle s_1, s_2 \rangle, c_1s_2 + c_2s_1 + s_1 \times s_2)\) and \((c, s)^{-1} = (c, -s)\) for unit norm \( q = (c, s)\). Substituting these in Eqn.\,(2) leads to the geodesic quaternion regression loss,

\[ \mathcal{L}_R(q_1, q_2) = 2 \cos^{-1}(|\langle q_1, q_2 \rangle|). \quad (8) \]

### 4.3 Geodesic Regression Model

We combine the network architecture and loss functions described earlier to form the Geodesic Regression network, which is trained while solving the optimization problem,

\[ \mathcal{R}_G : \min_W \frac{1}{N} \sum_n \mathcal{L}_R(y_n^*, y_n). \quad (9) \]

Here \( \mathcal{L}_R(y_n^*, y_n) \) is the geodesic loss (defined in Eqns.\,(5) and\,(8)) between ground-truth pose \( y_n^* \) and predicted pose \( y_n = \Phi_F(\Phi_F(x_n; W_F); W_{\Phi_F}) \) returned by the network given input image \( x_n \), and \( W = [W_F; \{ W_{\Phi_F} \}] \) includes all the parameters of the feature network and per-category regression pose networks.

### 4.4 Baselines

We compare the Geodesic Regression network with two common-sense baselines of (i) Euclidean Regression and (ii) Classification, which we describe next.

#### 4.4.1 Euclidean Regression Model

In this model, we train the network using the squared Euclidean loss instead of the Geodesic loss leading to the optimization problem,

\[ \mathcal{R}_E : \min_W \frac{1}{N} \sum_n \| y_n^* - y_n \|_2^2. \quad (10) \]

As we shall see later in §7.3 (Table 2) and §7.4 (Table 13), the Geodesic regression model is consistently better than the Euclidean regression model for both axis-angle and quaternion representations as it models the underlying geometry of the problem more faithfully.

#### 4.4.2 Classification Model

Here, we discretize the continuous pose space into bins defined by a given dictionary of key poses \( \{ z_k \in \mathbb{R}^3 \}_{k=1}^K \). Formally, given a pose \( y \in \mathbb{R}^3 \) (axis-angle representation), we assign it pose label \( l \in \{ 1, \ldots, K \} \) if key pose \( z_l \) is the closest to \( y \), i.e., \( l = \argmin_k \| y - z_k \|_2 \). The work of Li et al (2018) used a pre-specified dictionary of key poses obtained by a uniform tessellation of \( SO(3) \). In this work, we try to capture the geometry of the data (target orientation space) by learning a K-Means dictionary \( \{ z_k \} \) from training data as our key pose dictionary. Note that our pose target \( y \) can be any choice of orientation representation: axis-angle or quaternion representations. Then, the network architecture is the same as for the regression models with the difference being the last 2 layers of the pose network (shown in Fig. 4c). Notice that the output of the classification model is \( K \)-dimensional, where \( K \) is the size of the pose dictionary used to discretize the pose space and the final non-linearity is a softmax operation. The classification model is then trained by solving the optimization problem,

\[ \mathcal{C} : \min_W \frac{1}{N} \sum_n \mathcal{L}_C(l_n^*, l_n). \quad (11) \]

where \( \mathcal{L}_C(l_n^*, l_n) \) is the cross-entropy loss between ground-truth pose label,

\[ l_n^* = \argmin_k \| y_n^* - z_k \|_2, \quad (12) \]

and predicted pose label \( l_n = \Phi_F(\Phi_F(x_n; W_F); W_{\Phi_F}) \). The final pose output is given by \( y_n = z_{l_n} \).

### 5 Mixed Classification-Regression Networks for Orientation Estimation

As mentioned earlier, a disadvantage of the pure regression approach is that it is unable to properly model multimodal pose distributions arising in object categories
networks shown in §4, Fig. 3, the Bin & Delta model has two components: a Bin network (for classification) and a Delta network (for regression) as shown in Fig. 5. Both networks take as input the output of the feature network, \( f \). The Bin network predicts a pose-label \( l = \Phi_B(f; W^B) \), where \( \Phi_B \) is the Bin network parameterized by weights \( W^B \) associated with object category \( c \) and the pose-label \( l \) references a key pose \( z_l \). The Delta network predicts a pose-residual \( \delta y = \Phi_D(f; W^D) \) with Delta network \( \Phi_D \) and its weights \( W^D \). Given a classification output, pose label \( l \), and a regression output \( \delta y \) are combined as

\[
y = g(z_l, \delta y),
\]

where \( z_l \) is the key pose corresponding to label \( l \), \( \delta y \) is the deviation from that key pose and \( g(\cdot, \cdot) \) is some combination function. Therefore, the parameters of the Bin & Delta model are given by \( W = [W^B, \{W^D_l\}, \{W^D_z\}] \).

When designing the combination function two major choices arise. One choice is to have a single pose-residual network for all key poses, in which case the value of \( \delta y \) does not depend on the pose label \( l \). An alternative choice shown in Fig. 6 is to have a Delta model for every single pose-bin in which case, \( \delta y \rightarrow \delta y_l \) will also be a function of the pose label \( l \). This modeling decision is equivalent to deciding whether to have a common covariance matrix across all clusters or have a different covariance matrix for every cluster in a Gaussian Mixture Model (GMM).

5.2 Loss Functions

As implied by the name, the loss function for a Mixed Classification-Regression network is a combination of a regression loss and a classification loss. Specifically, the loss is given by

\[
\mathcal{L}(y_1, y_2) = \alpha \mathcal{L}_R(g(z_{l_1}, \delta y_{l_1}), g(z_{l_2}, \delta y_{l_2})) + \mathcal{L}_C(l_1, l_2),
\]

where we apply the geodesic regression loss \( \mathcal{L}_R \) on the pose outputs \( y_1 = g(z_{l_1}, \delta y_{l_1}) \) & \( y_2 = g(z_{l_2}, \delta y_{l_2}) \) and the classification loss \( \mathcal{L}_C \) on pose labels \( l_1 \) & \( l_2 \). The losses are combined with a relative weighting parameter \( \alpha \). The pose variables \( y_1 \) and \( y_2 \) can refer to any choice of orientation representation with an appropriately chosen combination function. The mixed loss of Eqn. (14) is an extension of the geodesic loss of Eqns. (5), (8) for the mixed representation with additional supervision on the classification component. Note that applying the regression loss at the pose output is also a modeling choice and as we shall see later in §5.3.6 and §5.3.7, we can instead apply a regression loss on the output of the Delta network to get a different model.

5.3 Models

Combining the network architecture and loss functions described above with different modeling choices leads to a variety of Bin & Delta models which are detailed in this section. An overview of all the models discussed here is provided in Table 1.
function of Eqn. (14), leads to the optimization problem

\[ \text{This choice of } g \text{ when substituted in the mixed-loss function of Eqn. (14), leads to the optimization problem} \]

\[ \mathcal{M}_G : \min W \sum_n \left[ \alpha \mathcal{L}_R(y_n^*, z_n + \delta y_n) + \mathcal{L}_C(l_n^*, l_n) \right], \quad (16) \]

where we minimize the weighted sum of two terms: (1) Geodesic loss \( \mathcal{L}_R \) between ground-truth pose \( y_n^* \) and predicted pose \( y_n = z_n + \delta y_n \), and (2) Cross-entropy loss \( \mathcal{L}_C \) between ground-truth pose label \( l_n^* \) (given by Eqn. (12)) and predicted pose label \( l_n \) returned by the Bin network.

A toy example of the Geodesic Bin & Delta model is illustrated in Fig. 7 in the context of points on a plane. Given training data (the points shown in red), K-Means clustering returns the dictionary of key poses: \( \{ z_1, z_2, z_3, z_1 \} \) (shown in blue +). A new point \( y \) (shown in a black *) with associated label \( l = 1 \) is now a sum of key pose \( z_1 \) and residual \( \delta y \) (shown in green). This is actually what is happening in the Geodesic Bin & Delta model too, except that the X-Y plane is now the tangent plane on the rotation manifold at the identity \( I_3 \) and the points are 3-dimensional instead of the 2-D points shown in the toy example in Fig. 7.

Using one Delta network per pose-bin (Fig. 6) instead of only one Delta network for all pose-bins (Fig. 5) in the Geodesic Bin & Delta model leads to the optimization problem

\[ \mathcal{M}_G^+ : \min W \sum_n \left[ \alpha \mathcal{L}_R(y_n^*, z_n + \delta y_n) + \mathcal{L}_C(l_n^*, l_n) \right], \quad (17) \]

where the modification is highlighted in red. This leads to more flexibility in modeling the pose residuals.

Another way to interpret the pose residual in the Geodesic Bin & Delta model is

\[ \delta y = \log y - z_l = \log(R) - \log(\tilde{R}_l), \quad (19) \]

where \( \tilde{R}_l = \exp(z_l) \) is the key rotation associated with key pose \( z_l \). As we mentioned earlier, the log operation projects the rotation matrix \( R \) onto the tangent plane at the identity, \( I_3 \), and Eqn. (19) defines the pose residual as a difference in the tangent plane. We propose an alternative definition of the pose residual

\[ \delta y = \log \tilde{R}_l(R), \quad (20) \]

where we now project the rotation matrix \( R \) onto the tangent plane at key rotation \( \tilde{R}_l \). This is equivalent to defining the combination function \( g(\cdot, \cdot) \) as

\[ g(z_l, \delta y) = \log(\exp(z_l) \exp(\delta y)), \quad (21) \]

We call this the Riemannian Bin & Delta model because we use the Riemannian exponential and logarithm maps in defining what the output of the Bin and Delta networks represent. The pose output is now given by

\[ R = \tilde{R}_l \exp(\delta y). \quad (22) \]

Substituting this into the mixed loss of Eqn. (14), we get the optimization problem

\[ \mathcal{M}_R : \min W \sum_n \left[ \alpha \mathcal{L}_R(R_n^*, \tilde{R}_l, \exp(\delta y_n)) + \mathcal{L}_C(l_n^*, l_n) \right], \quad (23) \]

where we again minimize the weighted sum of two terms: (1) Geodesic loss \( \mathcal{L}_R \) between ground-truth rotations \( R_n^* \) and predicted rotations \( \tilde{R}_l \), and (2) Cross-entropy loss \( \mathcal{L}_C \) between ground-truth class label \( l_n^* \) and predicted class label \( l_n \).

A toy example of the Riemannian Bin & Delta model for a circle is shown in Fig. 8. The figure shows a circle with 5 tangent planes corresponding to key poses \( \tilde{R}_i, i = 1, ..., 5 \). The rotation \( R \) (shown in red) is now a
combination of the key pose $\tilde{R}_1$, with pose label $l = 1$, and the delta $\delta y$ (shown in orange).

Again, using one Delta network per pose-bin we get the modified optimization problem

$$
\mathcal{M}_{R^+} : \min_w \frac{1}{N} \sum_n \left[ \alpha \mathcal{L}_R(R_n^*, \tilde{R}_1, \exp(\delta y_n^*) + \mathcal{L}_C(l_n^*, l_n) \right],
$$

(24)

where the difference is highlighted in red. Note that everything is defined in terms of rotations for the Riemannian Bin & Delta model and we use only the axis-angle representation in this model, which are easily transformed to rotations via the Rodrigues' rotation formula.

### 5.3.3 Probabilistic Bin & Delta Model

In the Geodesic Bin & Delta model, given the output probabilities of the Bin network for image $x_n$, $p_{nk}$, $k = \{1, \ldots, K\}$, we predict the most likely pose label as $l_n = \arg\max_k p_{nk}$ and use it during network training. A different formulation that better uses this probabilistic output is our Probabilistic Bin & Delta model which solves the optimization problem

$$
\mathcal{M}_P : \min_w \frac{1}{N} \sum_n \left[ \alpha \sum_k p_{nk} \mathcal{L}_R(y_n^*, z_k + \delta y_n) + \mathcal{L}_C(l_n^*, l_n) \right],
$$

(25)

where we are now weighting the geodesic loss between the ground-truth pose $y_n^*$ and per-category predicted pose $y_n^k = z_k + \delta y_n$ with the probability of the pose label being assigned that class, $p_{nk}$. The final predicted pose is still given by $y_n = z_{l_n} + \delta y_n$, where $l_n = \arg\max_k p_{nk}$, but we expect to learn better models due to more information in the modified optimization problem. The one-delta per pose-bin version of this model is given by the optimization problem

$$
\mathcal{M}_{P^+} : \min_w \frac{1}{N} \sum_n \left[ \alpha \sum_k p_{nk} \mathcal{L}_R(y_n^*, z_k + \delta y_n^k) + \mathcal{L}_C(l_n^*, l_n) \right],
$$

(26)

with the difference highlighted in red again.

### 5.3.4 RelaXed Bin & Delta

Another relaxation in our original problem formulation is that instead of a hard assignment (via K-Means), where we assign a single key-pose to an image i.e. $l_n^* = \arg\max_k \|y_n^* - z_k\|_2$, a more flexible model and possibly more informative model would be to do a soft assignment to all key-poses and use this probabilistic information in a better way. Post K-Means, one can generate a probabilistic assignment using

$$
p_n^k = \frac{\exp(-\gamma \|y_n^* - z_k\|_2^2)}{\sum_k \exp(-\gamma \|y_n^* - z_k\|_2^2)}.
$$

(27)

Now, the classification loss can be modified to be a Kullback-Leibler divergence $\mathcal{L}_{KD}$ between ground-truth and predicted probabilities. This modification in the Geodesic Bin & Delta model of Eqn. (16) leads to the RelaXed Bin & Delta model with the optimization

$$
\mathcal{M}_X : \min_w \frac{1}{N} \sum_n \left[ \alpha \mathcal{L}_R(y_n^*, z_{l_n} + \delta y_n) + \mathcal{L}_{KD}(p_n^*, p_n) \right],
$$

(28)

where $p_n^* = [p_{n,k}^*]_k=1^K$ and $p_{nk} = [p_{nk}]_k=1^K$ and its variant

$$
\mathcal{M}_{X^+} : \min_w \frac{1}{N} \sum_n \left[ \alpha \mathcal{L}_R(y_n^*, z_{l_n} + \delta y_n^k) + \mathcal{L}_{KD}(p_n^*, p_n) \right].
$$

(29)

### 5.3.5 RelaXed Probabilistic Bin & Delta Model

The same relaxation described in §5.3.4 can be applied to the Probabilistic Bin & Delta model to get RelaXed Probabilistic Bin & Delta models:

$$
\mathcal{M}_{XP} : \min_w \frac{1}{N} \sum_n \left[ \alpha \sum_k p_{nk} \mathcal{L}_R(y_n^*, z_k + \delta y_n) + \mathcal{L}_{KD}(p_n^*, p_n) \right],
$$

(30)

$$
\mathcal{M}_{XP^+} : \min_w \frac{1}{N} \sum_n \left[ \alpha \sum_k p_{nk} \mathcal{L}_R(y_n^*, z_k + \delta y_n^k) + \mathcal{L}_{KD}(p_n^*, p_n) \right].
$$

(31)

These were the models proposed as Probabilistic Bin & Delta previously (Mahendran et al., 2018), but as can
be seen above, they include two probabilistic changes and are studied separately as three different classes of models here.

5.3.6 Simple Bin & Delta Model

Instead of applying the regression loss at the pose output, we can apply it at the output of the Delta network leading to our Simple Bin & Delta model,

$$M_S : \min_{W} \frac{1}{N} \sum_{n} \left[ \alpha \| \delta y_n^* - \delta y_n \|^2_2 + \mathcal{L}_C(l_n^*, l_n) \right], \quad (32)$$

and its one-delta-per-pose-bin variant,

$$M_{S+} : \min_{W} \frac{1}{N} \sum_{n} \left[ \alpha \mathcal{L}_C(l_n^*, l_n) + \| \delta y_n^* - \delta y_n^{i+n} \|^2_2 \right]. \quad (33)$$

As discussed in Mahendran et al (2017), the geodesic loss function is highly non-convex with many local optima, which makes it important to initialize the network weights correctly. We initialize weights of the Geodesic models by training the Simple Bin & Delta model for weights correctly. We initialize weights of the Geodesic models by training the Simple Bin & Delta model for one epoch of training data. These Simple Bin & Delta models have been used in prior work (Li et al, 2018; Guler et al, 2017) and they explicitly enforce supervision on the outputs of the two individual networks.

5.3.7 Log-Euclidean Bin & Delta Model

The geodesic loss between the ground-truth and predicted rotations in the Riemannian Bin & Delta model can be approximated by the Euclidean distance on the tangent space at the identity,

$$\mathcal{L}(R_n^*, \tilde{R}_n \exp(\delta y_n)) = \mathcal{L}(\tilde{R}_n^T R_n^*, \exp(\delta y_n))$$

$$\approx \| \log(\tilde{R}_n^T R_n^*) - \log(\exp(\delta y_n)) \|^2_2. \quad (34)$$

This approximation is better the closer $R_n^*$ is to $R_n^*$ or alternately, the closer $\tilde{R}_n^T R_n^*$ is to the Identity. This new regression loss gives us the Log-Euclidean Bin & Delta model,

$$M_{LE} : \min_{W} \frac{1}{N} \sum_{n} \left[ \mathcal{L}_C(l_n^*, l_n) + \alpha \| \log(\tilde{R}_n^T R_n^*) - \delta y_n \|^2_2 \right],$$

and its variant,

$$M_{LE+} : \min_{W} \frac{1}{N} \sum_{n} \left[ \mathcal{L}_C(l_n^*, l_n) + \alpha \| \log(\tilde{R}_n^T R_n^*) - \delta y_n^{i+n} \|^2_2 \right]$$

where the term $\log(\tilde{R}_n^T R_n^*)$ can be precomputed for efficiency of training.

6 Data Augmentation using 3D Pose Jittering

We assume that each image is annotated with a 3D rotation $R(az, el, ct) = R_Z(ct) R_X(el) R_Z(az)$, where $R_Z$ and $R_X$ denote rotations around the z- and x-axis respectively. Jittered bounding boxes (bounding boxes with translational shifts that have sufficient overlap with the original box), like in V&K (Tulsiani and Malik, 2015), introduce small unknown changes in the corresponding $R$. Instead, we augment our data by generating new samples corresponding to known small shifts in camera-tilt and azimuth. We call this new augmentation strategy 3D pose jittering (see Fig. 9). Small shifts in camera-tilt lead to in-plane rotations, which are easily captured by rotating the image. Small shifts in azimuth or elevation angles lead to out-of-plane rotations, which can be described by homographies. We estimate these homographies in the following way: (i) we first project a CAD model of the object onto the image (we use a small percentage of the 3D points closest to the camera for the projection step), (ii) we rotate the object by a small angle in azimuth and/or elevation, (iii) we project the same set of points used earlier onto the image with new rotation (this is also the updated target for our training), and (iv) we compute a homography using the DLT algorithm (Hartley and Zisserman, 2004) between the two sets of projected points. We generate a dense grid of samples corresponding to $R(az \pm \delta az, el, ct \pm \delta ct)$. We also flip all samples, which corresponds to $R(-az, el, -ct)$.
7 Results and Discussion

First, we describe the Pascal3D+ dataset (Xiang et al., 2014), a popular benchmark dataset used for evaluating 3D pose estimation methods. Then we present the metrics we use to evaluate all our models. Then, we demonstrate the effectiveness of our framework and models with state-of-the-art performance on this challenging task. Finally, we present an ablation study on different decision choices we make in our pose estimation system.

7.1 Dataset

The Pascal3D+ consists of images of twelve object categories: aeroplane (aero), bicycle (bike), boat, bottle, bus, car, chair, diningtable (dtable), motorbike (nbike), sofa, train and tvmonitor (tv). These images were curated from the Pascal VOC 2012 Everingham et al (2015) and ImageNet Deng et al (2009) datasets, and annotated with 3D pose in terms of the Euler angles (az, el, ct). We use the ImageNet-trainval as our training data, Pascal-train images as our validation data and the Pascal-val images as our testing data. Following the protocol of Tulsiani and Malik (2015); Su et al (2015) and others, we use ground-truth bounding boxes of un-occluded and un-truncated objects. All our results in §7.3 are obtained using these ground-truth bounding boxes. We also evaluate the performance of our models on bounding boxes returned by object detection systems in §7.5.

7.2 Evaluation Metrics

To evaluate the performance of our models, we use two standard metrics proposed in Xiang et al (2014): MedErr and Acc_π. MedErr is the median angle error (in degrees) between ground-truth and predicted rotation,

\[
\text{MedErr} = \text{median}_{n=1}^N \angle(R^*_n, R_n),
\]

where \(\angle(R^*_n, R_n) = \cos^{-1}\left(\frac{\text{tr}(R^*_n R_n^-1)}{2}\right)\) is the angle between ground-truth rotation \(R^*_n\) and predicted rotation \(R_n\) for test image \(x_n\) (3). Acc_π is the percentage of test images that have angle error less than 30°

\[
\text{Acc}_\pi = \frac{1}{N} \sum_{n=1}^N \mathbf{1}[\angle(R^*_n, R_n) < 30^\circ].
\]

To evaluate the performance of 3D pose estimation models with detected bounding boxes instead of ground-truth ones, Xiang et al (2014) and Tulsiani and Malik (2015) extended the AP (Average Precision) metric popularly used in the object detection literature to two metrics: ARP_θ and AVP_K. While computing the AP metric, a detected bounding box is considered good if it has an intersection over union (IOU) overlap of at least 0.5 with a ground-truth box. For the ARP metric, the detected bounding box must have sufficient overlap IOU > 0.5 and have estimated rotation within 30° of the ground-truth rotation, \(\angle(R^*_n, R_n) < 30^\circ\). Note that we choose \(\theta = 30^\circ\) similar to Tulsiani and Malik (2015). For the AVP_K metric, the azimuth angles are binned into \(K\) non-overlapping bins and a detection is considered good if \(IOU > 0.5\) and \(\ell(az^*) = \ell(az)\) where \(\ell(az)\) is the bin corresponding to azimuth angle az. Previous works (Tulsiani and Malik, 2015; Su et al, 2015; Massa et al, 2016) have reported performance using the AVP_1, AVP_2, AVP_16 and AVP_24 metrics. We also report our performance on these numbers. However, note that this metric is unfair for our models because whereas all previous works enforce supervision on the Euler angles (the azimuth angle included) directly, we enforce supervision on rotation matrices. We compute the azimuth angle from our estimated rotations and compare with ground-truth.

7.3 3D Pose Estimation

We break down all our results along the lines of our models. We first discuss the performance of our pure regression models, then the classification model and finally our mixed-classification regression models. For all models discussed in this section and in §7.5, we use the axis-angle representation for rotation matrices. We discuss the choice of representation: axis-angle v/s quaternion as part of our ablation experiments in §7.4. For all experiments and models we discuss in the next sections, we ran each experiment three times and report the mean across these trials. For the error bars shown in all future figures, we use the standard deviation computed across these trials.

**Pure Regression.** Our first comparison is between the Geodesic and Euclidean regression models defined in §4.3 and §4.4.1 respectively. As can be seen in Table 2, using a geodesic regression loss significantly improves performance compared to Euclidean regression loss. Table 20 contains more detailed results with a breakdown per object category and the same behavior is observed for all object categories. This is in line with our expectations that a geodesic loss better reflects the underlying geometry of the problem and using this loss results in a big performance boost.

**Pure Classification.** We now evaluate the performance of the pure classification model described in §4.4.2. We ran K-Means clustering on the pose-targets of the ren-
Table 2: Performance of our Euclidean and Geodesic Regression models on the test set. Lower is better for the MedErr metric and higher is better for the Acc\_π metric.

| Model | MedErr | Acc\_π |
|-------|--------|--------|
| RE    | 15.50  | 0.7656 |
| RG    | 11.63  | 0.8166 |

Table 3: Performance of our Classification models on the validation set for different values of the size of the pose dictionary, K.

| K     | MedErr | Acc\_π |
|-------|--------|--------|
| 50    | 15.68  | 0.8127 |
| 100   | 13.64  | 0.8170 |
| 200   | 12.50  | 0.8206 |
| 400   | 12.85  | 0.8162 |

Table 4: Performance of our Classification model on the test set for K = 200 and a comparison with the Geodesic regression model.

| Model | MedErr | Acc\_π |
|-------|--------|--------|
| C     | 11.31  | 0.8298 |
| RG    | 11.63  | 0.8166 |

Table 5: Performance of our Geodesic Bin & Delta model M_G on the validation test for different values of relative weighting parameter, α, for K = 200.

| α     | MedErr | Acc\_π |
|-------|--------|--------|
| 0.1   | 12.83  | 0.8145 |
| 1.0   | 11.92  | 0.8212 |
| 10.0  | 13.90  | 0.8068 |

Table 6: Performance of the Geodesic Bin & Delta models, M_G(α = 1, K = 200) and M_G+(α = 10, K = 16), on the test set.

Fig. 10: Performance of the Geodesic Bin & Delta models per-category under the MedErr metric.
| Model | MedErr | Acc_π |
|-------|--------|--------|
| \(\mathcal{M}_R\) | 11.69  | 0.8285 |
| \(\mathcal{M}_R^+\) | 10.52  | 0.8573 |

Table 7: Performance of the Riemannian Bin & Delta models \(\mathcal{M}_R(\alpha = 1, K = 200)\) and \(\mathcal{M}_R^+ (\alpha = 10, K = 16)\), on the test set.

| K | \(\gamma\) | MedErr | Acc_π |
|---|---|--------|--------|
| 16 | 2.06 | 12.48 | 0.8302 |
| 50 | 7.82 | 12.98 | 0.8320 |
| 100 | 15.08 | 51.71 | 0.3932 |
| 200 | 25.23 | 46.73 | 0.4256 |

Table 9: Performance of the RelaXed Bin & Delta model \(\mathcal{M}_X\) on the validation set for different choices of the size of pose dictionary, \(K\).

Probabilistic Bin & Delta. These models (PBD in short) are described in §5.3.3 and are a probabilistic variation of the GBD models with the geodesic regression loss between ground-truth and predicted pose now weighted by the probability of pose-class predictions rather than just the most-likely pose-class. As can be seen in Table 8, these models perform slightly worse than the GBD models but are still competitive. The advantage of these models is that these did not require any initialization strategy unlike GBD where we needed to initialize the models with 1 epoch of training the Simple Bin & Delta models.

RelaXed Bin & Delta. These models (XBD in short) combine both the probabilistic weighting of geodesic losses and soft-assignment of pose-class labels. They were discussed in our previous work (Mahendran et al, 2018) as Probabilistic Bin & Delta mod-
Table 10: Performance of the RelaXed Probabilistic Bin & Delta models, \( \mathcal{M}_{XP}(\alpha = 1, K = 100, \gamma = 10) \) and \( \mathcal{M}_{XP} + (\alpha = 1, K = 16, \gamma = 10) \) on the test set.

| Model   | MedErr  | Acc \( \bar{\zeta} \) |
|---------|---------|------------------------|
| \( \mathcal{M}_{XP} \)  | 11.38   | 0.8185                 |
| \( \mathcal{M}_{XP}+ \) | 10.54   | 0.8470                 |

Table 11: Performance of our Simple Bin & Delta models, \( \mathcal{M}_S(\alpha = 1, K = 100) \) and \( \mathcal{M}_S + (\alpha = 1, K = 16) \), and Log-Euclidean Bin & Delta models, \( \mathcal{M}_{LE}(\alpha = 1, K = 100) \) and \( \mathcal{M}_{LE} + (\alpha = 1, K = 16) \) on the test set.

| Model   | MedErr  | Acc \( \bar{\zeta} \) |
|---------|---------|------------------------|
| \( \mathcal{M}_S \)        | 12.14   | 0.8303                 |
| \( \mathcal{M}_S+ \)       | 11.95   | 0.8387                 |
| \( \mathcal{M}_{LE} \)     | 12.11   | 0.8410                 |
| \( \mathcal{M}_{LE}+ \)    | 11.99   | 0.8329                 |

7.4 Ablation Analysis

In this section, we discuss three decision choices we made in our pose estimation pipeline: (1) Choice of representation, (2) Choice of feature network and (3) Choice of data augmentation.

**Orientation Representation.** As we discussed earlier in §3.1, we can represent a rotation matrix in terms of Euler angles, Axis-angles or Quaternions. We prefer to use the axis-angle representation due to its compactness and the geodesic properties defined on top of it. However, quaternions could also have been chosen and all the models re-defined with them instead. We now study how the choice of quaternions as our pose representation would affect the performance of our models. We use quaternions in our Euclidean regression model \( \mathcal{R}_E \), our Geodesic Regression model \( \mathcal{R}_G \) and our Geodesic Bin & Delta models \( \mathcal{R}_G \) and \( \mathcal{R}_{G+} \). As can be seen in Table 13 and Fig. 14, both representations are equivalent for most object categories with axis-angle being better for some. For some models like the Riemannian Bin & Delta and the Log-Euclidean Bin & Delta, the axis-angle representation is a more direct choice compared to quaternions.

**Feature Network.** In our first work on 3D pose regression (Mahendran et al, 2017), we used the VGG-M network (Chatfield et al., 2014) and in our recent work, we use the ResNet50 network (He et al., 2016a). One could exhaustively search over all possible feature networks and find the one that performs the best but that is not the aim of this work. Instead, we just analyze two choices of feature networks: a standard VGG13 with Batch-Norms (from the PyTorch model zoo) and a ResNet50 (also from the PyTorch model zoo) and compare their performance under our Geodesic Bin & Delta models. As can be seen in Table 14, the models with a ResNet feature network perform better than the models with a VGG feature network. This has also been observed previously in other works (Grabner et al., 2018).

**Data Augmentation.** We use both 3D Pose jittered and rendered images to augment our training data. We now study how using just one of these would affect the
### Table 12: Comparison with current state-of-the-art algorithms for 3D pose estimation from 2D images on the Pascal3D+ dataset under different metrics. Lower is better for the MedErr metric and higher is better for the Accuracy metric. Best results are highlighted in bold and second-best results are shown in red (best seen in color).

| Metric | Model | Mean |
|--------|-------|------|
| MedErr | (Tulsiani and Malik, 2015) | 13.59 |
|       | (Su et al., 2015) | 11.68 |
|       | (Mousavian et al., 2017) | 11.15 |
|       | (Grabner et al., 2018) | 10.88 |
|       | MG+ | 10.10 |
|       | MR+ | 10.52 |
|       | MP+ | 10.80 |
|       | MXP+ | 10.54 |

| Metric | Model | Mean |
|--------|-------|------|
| Accπ | (Tulsiani and Malik, 2015) | 0.8075 |
|       | (Su et al., 2015) | 0.8200 |
|       | (Mousavian et al., 2017) | 0.8103 |
|       | (Grabner et al., 2018) | 0.8592 |
|       | MG+ | 0.8588 |
|       | MR+ | 0.8573 |
|       | MP+ | 0.8457 |
|       | MXP+ | 0.8470 |

### Table 13: Performance of Axis-angle and Quaternion pose representations on the test set under different models.

| Model | Representation | MedErr | Accπ |
|-------|----------------|--------|------|
| RE | Axis-angle | 15.50 | 0.7656 |
| RG | Quaternion | 14.14 | 0.7965 |
| MG | Axis-angle | 11.63 | 0.8166 |
| MR | Quaternion | 12.20 | 0.8141 |
| MP | Axis-angle | 11.44 | 0.8439 |
| MR | Quaternion | 11.23 | 0.8384 |
| MG+ | Axis-angle | 10.10 | 0.8588 |
| MR | Quaternion | 10.75 | 0.8560 |

### Fig. 13: Comparison of our models with state-of-the-art pose estimation methods under different metrics. Current state-of-the-art models are shown in red. The pure regression and classification models are shown in green. Our Bin & Delta models are shown in blue. Lower is better.

### Fig. 14: Comparison between Axis-angle and Quaternion representations under the MedErr metric.
Table 14: Comparison between ResNet50 and VGG13 feature networks with the Geodesic Bin & Delta models.

| Model  | Network   | MedErr | Acc_{\pi} |
|--------|-----------|--------|-----------|
| M_G    | ResNet50  | 11.44  | 0.8439    |
|        | VGG13     | 11.83  | 0.8277    |
| M_G+   | ResNet50  | 10.10  | 0.8588    |
|        | VGG13     | 10.81  | 0.8423    |

Table 15: Comparison between different types of data augmentation with the Geodesic Bin & Delta models.

| Model  | Data    | MedErr | Acc_{\pi} |
|--------|---------|--------|-----------|
| M_G    | Rendered| 14.79  | 0.7779    |
|        | Augmented| 18.35  | 0.6757    |
|        | Both    | 11.44  | 0.8439    |
| M_G+   | Rendered| 15.03  | 0.7748    |
|        | Augmented| 14.30  | 0.7506    |
|        | Both    | 10.10  | 0.8588    |

Table 16: Performance of our Geodesic Bin & Delta models on the test using under the ARP\_{\pi} metric with different detected bounding boxes. Higher is better.

| Model  | Detections | ARP\_{\pi} | AP   |
|--------|------------|------------|------|
| V&K    | V&K        | 51.6       | 61.6 |
| M_G    | V&K        | 52.6       | 61.0 |
| M_G+   | R4CNN      | 51.3       | 61.0 |
| M_G    | Mask-RCNN  | 61.6       | 76.2 |
| M_G+   | Mask-RCNN  | 63.9       | 76.2 |

3D Pose Estimation with Detected bounding boxes

So far all the experimental evaluation was done with ground-truth bounding boxes with un-occluded and un-truncated objects. This allows us to focus on the orientation estimation problem assuming known detection and categorization. In this section, instead of ground-truth bounding boxes returned by an oracle, we use bounding boxes returned by an object detection system. As mentioned earlier in §7.2, we compare with other methods under the modified metrics for joint object detection and pose estimation: the ARP\_{\pi} and AVP_{K} metrics where K = 4, 8, 16, 24. We again emphasize that the AVP metric is an unfair metric for our models because we do not enforce any supervision on the azimuth angle unlike other models. We still report these numbers for the sake of completeness.

Firstly, as can be seen in Table 16, we significantly improve upon the state-of-the-art (Tulsiani and Malik, 2015) under the ARP\_{\pi} metric. We evaluate the performance of our Geodesic Bin & Delta models (M_G and M_G+) with bounding boxes returned by three object detection systems: (i) RCNN\^{2} (Tulsiani and Malik, 2015) (ii) RCNN\^{3} (Su et al, 2015) and (iii) Mask RCNN\^{4} (He et al, 2017; Girshick et al, 2018). A direct comparison between our work and that of Tulsiani and Malik (2015) using their detected bounding boxes shows that we improve upon their performance by at least 5 points. This shows that we are actually improving the pose estimation performance and not just boosting the metric artificially using better object detection systems. Using the current state-of-the-art detection system of Mask RCNN leads to an even bigger improvement of around 15.1 points for the M_G model and 17.4 points for the M_G+ model. Additionally, we observe that the M_G+ model is consistently better than the M_G model for all detection systems. This is consistent with the performance on ground-truth boxes where the M_G+ model (11.0^\circ) has lower median angle error compared to the M_G model (11.44^\circ). Similarly we observe that the ARP performance follows the order RCNN < V&K < Mask-RCNN. This is also consistent with the object detection performance under the AP metric. We analyze the performance of these models in more detail in Table 17 and it shows that better detections is just part of the reason for improved performance. We report three metrics: (i) % Detected: the percentage of bounding boxes detected correctly by the detection system (IoU overlap > 0.5), (ii) % Correct: the percentage of bounding boxes detected correctly and having pose error < 30^\circ, and (iii) MedErr: median pose error (in degrees) for all detected bounding boxes. Note that Pose-Err includes even those cases where the bounding box was detected correctly but the pose was estimated incorrectly. From this table, we see that both % Detected and % Correct increase with a better detection system but also the Pose-Err goes down. This tells us that we are not only detecting more bounding boxes with Mask-RCNN + M_G+ but also estimating their pose more accurately.

As can be seen in Table 18, for the AVP metric, we are clearly better than Su et al (2015) but are worse than Tulsiani and Malik (2015) and Massa et al (2016), especially for a higher number of azimuth bins K = 24. Note that Massa et al (2016) solves a joint object detection and pose estimation problem within a Fast-RCNN framework.

\^{2} http://www.cs.berkeley.edu/~shubhtuls/cachedir/vpsKps/VOC2012_val_det.mat
\^{3} https://github.com/ShapeNet/RenderForCNN/tree/master/data/detection_results/rcnn_bbox_reg_pruned
\^{4} X-101-64x4d-FPN backbone (model #36494496): https://github.com/facebookresearch/Detectron/blob/master/MODEL_ZOO.md
### 8 Conclusion

Orientation estimation is a challenging computer vision problem with applications in many domains. Current deep learning methods, in spite of their impressive performance, do not fully exploit the geometry of the orientation space. We designed regression-based deep learning models that use representations and loss functions that respect the Riemannian structure of the orientation space. We also built a framework of mixed classification-regression that leads to a family of Bin & Delta models with detected bounding boxes. For every object category, the Bin network is of size 2048-1000-500-200 for choice of $K = 100$ and the Delta network is of size 2048-1000-500-3. For the One-delta-per-bin models: $\mathcal{M}_{G+}$, $\mathcal{M}_{R+}$, $\mathcal{M}_{P+}$ and $\mathcal{M}_{XP+}$, for every object category, the Bin network is of size 2048-1000-500-16 for choice of $K = 16$ and the Delta network is of size of 16 x 2048-100-3 as we have 16 delta networks in total (1 per bin). Different object categories have different numbers of images in the Pascal3D+ dataset and we use a sampling strategy to balance the data in every mini-batch. We sample 4 rendered images and 4 real (augmented) images per object category in every mini-batch leading to a batch size of 96 images. This was the largest batch size that fit into a single GPU (Titan Xp) and was chosen for all One-bin-and-delta models. For the One-delta-per-bin models, we used 3 rendered images and 3 real images per object category instead as the largest batch size that fit into a single GPU.

We generated augmented data with azimuth shifts $[-1, 0, 1]$, elevation shifts $[-1, 0, 1]$ and camera-tilt shifts $[-4, -2, 0, 2, 4]$.

### Acknowledgements

This research work was supported by NFS grants 1527340 and 1834427.

### B More Results

In §7.3 and §7.4, we provided results for different models and experiments averaged across all twelve object categories. We now provide expanded tables with per-category results. We also provide a list of correspondences in Table 19.

| Condensed Results | Expanded Results |
|-------------------|-----------------|
| Table 2           | Table 20        |
| Table 3           | Table 21        |
| Table 4           | Table 22        |
| Table 5           | Table 23        |
| Tables 6,7,8,10,11| Table 24        |
| Table 9           | Table 25        |
| Table 13          | Table 26        |
| Table 15          | Table 27        |
| Table 14          | Table 28        |
| Table 16          | Tables 29,30    |
| Table 17          | Table 31        |
| Table 18          | Table 32        |

Table 19: An overview of all tables in Appendix B.

### References

Aubry M, Maturana D, Efros AA, Russell BC, Sivic J (2014) Seeing 3D Chairs: Exemplar Part-Based 2D-3D Alignment

Girshick, R (2015) detection framework, which in principle should give better results than systems trained on ground-truth bounding boxes and evaluated on detected ones (like V&K, R4CNN and Ours)

Mahendran et al.
Table 20: Comparison between the Euclidean and Geodesic regression models under two metrics. Lower is better for the MedErr metric and higher is better for the Acc₂ metric. Best results are in bold.

| Metric | Model | Dict. Size | MedErr | Acc₂ |
|--------|-------|------------|--------|------|
|        |       |            | R₀     | C   |
| 50     |       | 14.9       | 16.6   | 0.84 |
| 100    |       | 13.3       | 16.1   | 0.86 |
| 200    |       | 10.7       | 16.5   | 0.87 |
| 400    |       | 10.3       | 16.9   | 0.86 |

Table 21: Performance of the pure classification models on the validation set under two metrics for different dictionary sizes. Lower is better for the MedErr metric and higher is better for the Accuracy metric. Best results are in bold.

| Metric | Model | Dict. Size | MedErr | Acc₂ |
|--------|-------|------------|--------|------|
|        |       |            | R₀     | C   |
| 14.5   |       | 17.7       | 39.3   | 0.82 |
| 11.8   |       | 15.9       | 27.2   | 0.77 |
| 10.7   |       | 16.5       | 23.7   | 0.80 |
| 10.3   |       | 16.9       | 24.6   | 0.80 |

Table 22: Performance of the pure classification model on the test set with dictionary size $K = 200$. Lower is better for the MedErr metric and higher is better for the Accuracy metric.
| Metric | α | MedErr | Mean |
|-------|---|-------|------|
|       |   |       |      |
|       | 0.1 | 11.0 | 16.1 | 23.8 | 8.7 | 3.4 | 5.8 | 12.2 | 30.1 | 13.6 | 9.1 | 6.8 | 13.3 | 12.83 |
|       | 1.0 | 11.0 | 15.0 | 22.7 | 8.2 | 3.6 | 6.0 | 12.1 | 21.0 | 13.4 | 9.7 | 6.2 | 13.2 | 11.92 |
|       | 10.0 | 11.4 | 16.6 | 23.9 | 8.3 | 3.4 | 5.6 | 12.4 | 40.3 | 13.8 | 11.0 | 6.7 | 13.1 | 13.90 |
|       | 0.1 | 0.86 | 0.77 | 0.57 | 0.92 | 0.97 | 0.92 | 0.78 | 0.51 | 0.83 | 0.89 | 0.87 | 0.88 | 0.8145 |
|       | 1.0 | 0.87 | 0.75 | 0.57 | 0.93 | 0.97 | 0.93 | 0.80 | 0.55 | 0.83 | 0.91 | 0.86 | 0.89 | 0.8212 |
|       | 10.0 | 0.86 | 0.77 | 0.55 | 0.93 | 0.97 | 0.93 | 0.78 | 0.44 | 0.80 | 0.93 | 0.85 | 0.88 | 0.8968 |

Table 23: Performance of the Geodesic Bin & Delta model $\mathcal{M}_G$ with dictionary size 200 on the validation set for different choices of weighting parameter $\alpha$. Lower is better for the MedErr metric and higher is better for the Accuracy metric.

| Metric | Model | MedErr | Mean |
|-------|-------|--------|------|
|       | $\mathcal{M}_S$ |       |      |
|       | $\mathcal{M}_{S+}$ |       |      |
|       | $\mathcal{M}_G$ |       |      |
|       | $\mathcal{M}_{G+}$ |       |      |
|       | $\mathcal{M}_R$ |       |      |
|       | $\mathcal{M}_{R+}$ |       |      |
|       | $\mathcal{M}_{LE}$ |       |      |
|       | $\mathcal{M}_{LE+}$ |       |      |
|       | $\mathcal{M}_P$ |       |      |
|       | $\mathcal{M}_{P+}$ |       |      |
|       | $\mathcal{M}_{XP}$ |       |      |
|       | $\mathcal{M}_{XP+}$ |       |      |

Table 24: Performance of the Bin & Delta models on the test set under different metrics. Lower is better for the MedErr metric and higher is better for the Accuracy metric. Best results are highlighted in bold and second-best results are shown in red (best seen in color).

| Metric | K | γ | MedErr | Mean |
|-------|---|---|--------|------|
|       |   |   |        |      |
|       | 16 | 2.06 | 12.7 | 16.9 | 24.4 | 8.0 | 3.1 | 5.7 | 13.2 | 16.0 | 16.3 | 12.6 | 6.7 | 14.1 | 12.48 |
|       | 50 | 7.82 | 12.4 | 17.4 | 26.3 | 9.0 | 3.1 | 6.1 | 13.4 | 16.6 | 17.7 | 13.7 | 6.4 | 13.7 | 12.98 |
|       | 100 | 15.08 | 86.0 | 81.8 | 75.6 | 11.6 | 23.8 | 77.3 | 84.6 | 23.6 | 70.4 | 45.3 | 17.0 | 23.4 | 51.71 |
|       | 200 | 25.23 | 75.0 | 87.0 | 81.2 | 11.0 | 19.2 | 46.9 | 80.2 | 22.2 | 58.5 | 43.0 | 13.1 | 23.4 | 46.73 |

Table 25: Performance of the RelaXed Bin & Delta models $\mathcal{M}_X$ on the validation set under two metrics for different dictionary sizes. Lower is better for the MedErr metric and higher is better for the Accuracy metric.
Table 26: Comparison between the Axis-angle and Quaternion representations for different 3D pose estimation models.

Table 27: Performance of the Geodesic Bin & Delta models on the test set with different types of training data. Lower is better for the MedErr metric and higher is better for the Accuracy metric.

Table 28: Performance of the Geodesic Bin & Delta models on the test set using different feature networks. Lower is better for the MedErr metric and higher is better for the Accuracy metric.

Table 29: Performance of our Geodesic Bin & Delta models under the ARP metric using detected bounding boxes. We use the bounding boxes provided by V&K (Tulsiani and Malik, 2015) and R4CNN (Su et al, 2015). Higher is better.
| Model | Detection | Metric | V&K | R4CNN | Mask-RCNN | M_G | M_G+ | GT # BBoxes |
|-------|-----------|--------|-----|-------|-----------|------|------|-------------|
|       |           |        |     |       |           |      |      | 433 358 424 630 301 1004 1176 305 356 285 315 392 |
|       |           |        |     |       |           |      |      | 0.69 0.58 0.32 0.52 0.69 0.58 0.35 0.32 0.61 0.57 0.68 0.66 |
|       |           |        |     |       |           |      |      | 9.8 18.0 |
|       |           |        |     |       |           |      |      | 0.7258 |
|       |           |        |     |       |           |      |      | 0.5347 |
|       |           |        |     |       |           |      |      | 0.8158 |
|       |           |        |     |       |           |      |      | 0.7392 |
|       |           |        |     |       |           |      |      | 13.83 |

Table 30: Performance of the detected bounding boxes under the AP metric. Higher is better.

Table 31: Performance of the Geodesic Bin & Delta models with detected bounding boxes. We report three metrics for each combination of model and detection system: (i) % Detected: the percentage of bounding boxes detected correctly by the detection system (IoU overlap > 0.5), (ii) % Correct: the percentage of bounding boxes detected correctly and having pose error < 30°, and (iii) MedErr: median pose error (in degrees) for all detected bounding boxes. Higher is better for the percentages and lower is better for Pose-Err.

Hejrati M, Ramanan D (2014) Analysis by Synthesis: 3D Object Recognition by Object Reconstruction. In: 2014 IEEE Conference on Computer Vision and Pattern Recognition, pp 2449–2456, DOI 10.1109/CVPR.2014.314
Hou B, Miolane N, Khanal B, Lee MC, Alansary A, McDonagh S, Hajnal JV, Rueckert D, Glocker B, Kainz B (2018) Computing CNN Loss and Gradients for Pose Estimation with Riemannian Geometry. coRR abs/180501026
Jordan MJ, Jacobs RA (1994) Hierarchical Mixtures of Experts and the EM Algorithm. Neural Computation 6(2):181–214, DOI 10.1162/neco.1994.6.2.181
Kendall A, Cipolla R (2016) Modelling uncertainty in deep learning for camera relocalization. In: 2016 IEEE International Conference on Robotics and Automation (ICRA), pp 4762–4769, DOI 10.1109/ICRA.2016.7487679
Kendall A, Cipolla R (2017) Geometric Loss Functions for Camera Pose Regression with Deep Learning. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 6555–6564, DOI 10.1109/CVPR.2017.694
Kendall A, Grimes M, Cipolla R (2015) PoseNet: A convolutional network for real-time 6-dof camera relocalization. In: 2015 IEEE International Conference on Computer Vision (ICCV), pp 2938–2946, DOI 10.1109/ICCV.2015.336
Li C, Bai J, Hager GD (2018) A Unified Framework for Multi-View Multi-Class Object Pose Estimation. coRR abs/180108103
Liebelt J, Schmid C (2010) Multi-view object class detection with a 3D geometric model. In: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp 1688–1695, DOI 10.1109/CVPR.2010.5539836
Lim JJ, Pirsiavash H, Torralba A (2013) Parsing ikea objects: Fine pose estimation. In: 2013 IEEE International Conference on Computer Vision, pp 2992–2999, DOI 10.1109/ICCV.2013.372
Lim JJ, Khosla A, Torralba A (2014) FPM: Fine Pose Parts-Based Model with 3D CAD Models. In: Fleet D, Pajdla T, Schiele B, Tuytelaars T (eds) Computer Vision – ECCV 2014, Springer International Publishing, Cham, pp 478–493
Lopez-Sastre RJ, Tuytelaars T, Savarese S (2011) Deformable part models revisited: A performance evaluation for object category pose estimation. In: 2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops), pp 1052–1059, DOI 10.1109/ICCVW.2011.6130367
Ma Y, Soatto S, Kosecka J, Sastry S (2003) An Invitation to 3D Vision: From Images to Geometric Models. Springer Verlag
Mahendran S, Ali H, Vidal R (2017) 3D Pose Regression Using Convolutional Neural Networks. In: 2017 IEEE International Conference on Computer Vision Workshop (ICCV Workshops), pp 2174–2182, DOI 10.1109/ICCV.2017.254
Mahendran S, Ali H, Vidal R (2018) A Mixed Classification-Regression Framework for 3D Pose Estimation from 2D Images. coRR abs/180503225
Massa F, Aubry M, Marlet R (2014) Convolutional Neural Networks for joint object detection and pose estimation: A
| # Bins | Model | Detections | detected | bike | chair |  | mbike | sofa | train | tv | Mean |
|-------|-------|------------|----------|------|-------|-----|-------|------|-------|---|------|
| 4     | V&K   | 61.0 59.7 24.3 - | 68.1 57.7 29.3 | 22.9 | 61.2 47.6 61.7 | 57.1 49.9 |
|       | R4CNN | 63.4 61.2 23.4 - | 63.6 53.5 24.6 | 25.3 | 63.2 47.1 54.5 | 63.5 49.4 |
|       | Mask-RCNN | 74.9 69.8 41.2 - | 77.6 55.9 50.1 | 40.5 | 74.1 55.3 72.2 | 70.1 62.0 |
| 8     | V&K   | 57.5 54.8 18.9 - | 59.4 51.3 24.7 | 20.4 | 59.5 43.7 53.3 | 45.6 44.5 |
|       | R4CNN | 44.5 41.1 10.1 - | 48.0 36.6 13.7 | 15.1 | 39.9 26.8 39.1 | 46.5 32.9 |
|       | Mask-RCNN | 66.0 62.5 31.2 - | 68.7 55.7 19.2 | 31.9 | 64.0 44.7 61.8 | 58.0 51.3 |
|       | 4G    | 47.8 51.5 17.1 - | 52.2 46.6 25.7 | 22.4 | 53.2 42.7 51.1 | 45.9 41.5 |
|       | R4CNN | 49.9 50.7 14.8 - | 50.2 43.6 20.2 | 24.2 | 54.9 37.9 45.2 | 51.1 40.2 |
|       | Mask-RCNN | 58.3 59.0 30.2 - | 60.0 42.5 42.7 | 35.2 | 64.0 46.8 60.7 | 57.4 50.6 |
| 16    | V&K   | 51.4 43.0 23.6 - | 68.9 46.3 15.2 | 29.3 | 49.4 35.6 47.0 | 37.3 40.6 |
|       | R4CNN | 53.9 53.5 12.6 - | 48.2 35.3 16.0 | 18.2 | 31.9 27.7 30.5 | 34.6 29.2 |
|       | Mask-RCNN | 44.0 38.1 16.5 - | 53.5 33.7 33.1 | 25.3 | 37.6 33.3 45.0 | 40.8 36.4 |
|       | 4G    | 43.3 34.7 13.5 - | 50.4 40.5 23.2 | 15.1 | 36.9 33.0 41.1 | 33.0 33.1 |
|       | R4CNN | 44.2 35.1 12.2 - | 47.1 40.4 16.9 | 16.4 | 38.6 28.8 33.2 | 38.6 32.0 |
|       | Mask-RCNN | 51.1 41.7 19.0 - | 57.6 37.3 35.5 | 25.8 | 47.6 36.7 48.0 | 41.5 40.2 |
|       | 4K    | 37.0 33.4 10.0 - | 54.1 40.0 17.5 | 19.9 | 34.3 28.9 33.9 | 22.7 31.4 |
|       | R4CNN | 21.5 22.0 4.1 - | 38.6 25.5 7.4 | 11.0 | 24.4 15.0 28.0 | 19.8 19.8 |
|       | Mask-RCNN | 43.2 39.4 16.8 - | 61.0 44.2 13.5 | 29.4 | 37.5 33.5 46.6 | 32.5 36.1 |
| 24    | V&K   | 26.0 26.8 8.2 - | 38.0 31.2 16.7 | 12.9 | 26.4 30.2 36.6 | 25.1 25.2 |
|       | R4CNN | 28.9 24.7 7.4 - | 38.6 29.5 13.3 | 12.7 | 28.2 24.2 33.0 | 27.5 24.4 |
|       | Mask-RCNN | 31.3 28.2 14.8 | 42.1 26.0 26.3 | 19.9 | 35.4 30.5 44.0 | 31.4 30.0 |
|       | 4G    | 32.9 26.5 10.4 - | 42.7 37.6 18.5 | 13.6 | 29.5 27.5 37.3 | 23.9 27.3 |
|       | R4CNN | 35.0 25.5 7.7 - | 37.7 36.2 14.2 | 16.4 | 30.7 25.4 33.3 | 27.5 26.3 |
|       | Mask-RCNN | 40.4 33.7 16.0 - | 49.3 32.1 29.3 | 20.5 | 38.2 31.6 44.3 | 35.0 33.5 |

Table 32: Performance of our Geodesic Bin & Delta models under the AVP metric using detected bounding boxes. We compare with V&K (Tulsiani and Malik, 2015), R4CNN (Su et al, 2015) and Massa (Massa et al, 2016). We use the bounding boxes provided by V&K (Tulsiani and Malik, 2015) and R4CNN (Su et al, 2015). Higher is better. Note that our models do not have any supervision on the azimuth angles unlike the other methods which use cross-entropy loss on discretized azimuth angle bins.
ference on Computer Vision (ICCV), pp 2686–2694, DOI 10.1109/ICCV.2015.308

Tulsiani S, Malik J (2015) Viewpoints and keypoints. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 1510–1519, DOI 10.1109/CVPR.2015.7298758

Tulsiani S, Gupta S, Fouhey D, Efros AA, Malik J (2018) Factoring Shape, Pose, and Layout from the 2D Image of a 3D Scene. In: IEEE Conference on Computer Vision and Pattern Recognition

Wang Y, Li S, Jia M, Liang W (2016) Viewpoint estimation for objects with convolutional neural network trained on synthetic images. In: Chen E, Gong Y, Tie Y (eds) Advances in Multimedia Information Processing - PCM 2016, Springer International Publishing, Cham, pp 169–179

Wang Y, Tan X, Yang Y, Liu X, Ding E, Zhou F, Davis LS (2018) 3D Pose Estimation for Fine-Grained Object Categories. coRR abs/180604314

Wu J, Xue T, Lim JJ, Tian Y, Tenenbaum JB, Torralba A, Freeman WT (2016) Single Image 3D Interpreter Network. In: Leibe B, Matas J, Sebe N, Welling M (eds) Computer Vision – ECCV 2016, Springer International Publishing, Cham, pp 365–382

Xiang Y, Mottaghi R, Savarese S (2014) Beyond PASCAL: A benchmark for 3D object detection in the wild. In: IEEE Winter Conference on Applications of Computer Vision, pp 75–82, DOI 10.1109/WACV.2014.6836101