

Estimating small differences in car-pose from orbits

Berkay Kicanaoglu
Ran Tao
Arnold W.M. Smeulders
{b.kicanaoglu,r.tao,a.w.m.smeulders}@uva.nl

QUVA Lab,
University of Amsterdam,
the Netherlands

Abstract

Distinction among nearby poses and among symmetries of an object is challenging. In this paper, we propose a unified, group-theoretic approach to tackle both. Different from existing works which directly predict absolute pose, our method measures the pose of an object relative to another pose, *i.e.*, the pose difference. The proposed method generates the complete orbit of an object from a single view of the object with respect to the subgroup of $SO(3)$ of rotations around the z -axis, and compares the orbit of the object with another orbit using a novel orbit metric to estimate the pose difference. The generated orbit in the latent space records all the differences in pose in the original observational space, and as a result, the method is capable of finding subtle differences in pose. We demonstrate the effectiveness of the proposed method on cars, where identifying the subtle pose differences is vital.

1 Introduction

While pose estimation has recently gained substantial progress [10, 19, 20], distinction among nearby poses and among distant symmetries of the object remain a hard problem [10]. At the same time, pose estimation is still an important problem. In traffic, where pose determines the future direction of the car, it is even vital.

Confusion among nearby poses will demonstrate itself at many places but subtly. When the object resembles a tube, no difference in pose can be observed perpendicular to the main tube-axis due to rotational symmetry. When in frontal view or when in view at the round corners of a car, the car may look similar to such a tube. For these types of views, finding pose differences will be hard or very hard, see Figure 1. In general, the distinction of nearby poses rests on the accumulation of subtle differences distributed over the field of view.

Confusion among symmetries is due to the overall geometry of objects. To discriminate the right side of a car from its left side rests on the detection of sparsely distributed details of difference between the two sides. The distinction of the front of a car from its back requires the detection of sparsely distributed small local differences.

In traffic, exactly these hard conditions are crucial. Distinction among nearby poses is vital to establish the intent of the opposing car to cross before us. And, distinction among symmetries in a split second is vital to determine whether the opposing car is moving towards or away from us. In many cases, for example when following an opposing car in a sequence, changes in pose *relative to its previous pose* are important to detect immediately.



Figure 1: The hardness of distinction among nearby poses.

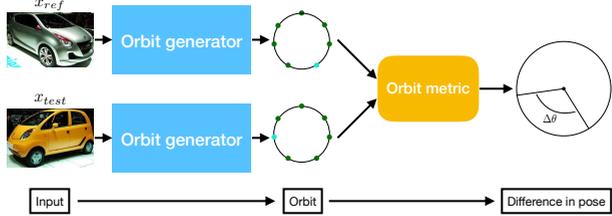


Figure 2: Our approach generates an orbit for each input starting from its initial pose, then compares two orbits to determine the number of steps of rotation necessary to align test object’s pose to that of the reference object using the orbit metric.

In this work, we take one unified, group-theoretic approach to merge both nearby and symmetry pose distinction into one approach. We construct an *equivariant, irreducible and interpretable* latent representation to encode the rotation faithfully. Central to our approach is the *orbit* [14], which is the ordered point set representing all transformed states of a given object with respect to the transformation at hand. In our case, we consider $SO(3)$, *i.e.* the 3D-rotation group.

From this foundation, we propose to generate the orbit per object. We do so in the latent space by learning to rotate them with respect to the subgroup of $SO(3)$ of rotations around the z -axis. The orbit records all relative relations in pose between any pair of views. This is relevant as the network is learning how to distinguish among any pair of poses, thus employing all possible information, from nearby, intermediate or distant poses alike. While current methods predict absolute pose [2, 11, 19, 20] (*Is this a 50-degree pose?*), our method measures the pose of an object relative to another pose, *i.e.*, the pose difference (*What is the difference between the two poses?*). Absolute pose can be established by gauging it with another known pose. But in many case it is important only to measure subtle relative pose differences.

This paper makes the following contributions. (1) We approach pose estimation from group theory by considering the orbit with appropriate constraints including an orbit-based metric to measure pose difference. (2) A novel network is proposed to generate the orbit of an object from a single view of the object after a learning phase from multiple views with known poses of other objects. (3) We undercut the necessary large amount of data by a learning strategy using synthetic as well as real data.

In our experiments we will evaluate on the basis of absolute pose, as is common to do, noting that relative pose difference (for example comparing with the previous view) would reveal the qualities of the method more favorably. We evaluate the method on car images as there the two hard conditions are vital: distinction among nearby poses and distinction among symmetries of an object. We observe that among similar network-models we achieve top-performance on these hard but important cases.

2 Related Work

Pose estimation has been treated as a regression task in [9] from local descriptors of the same patch in different viewpoints. [10] uses probabilistic regression based on Hough Forests

with an uncertainty criterion for continuous pose estimation. Similarly, [6] uses regressors on Fisher-encoded vectors extracted from spatial cells. [7] trains a separate SVM-classifier per viewpoint bin using off-the-shelf CNN-features. [10, 19] adopt data-driven classification by CNNs. In [19], the authors propose a rendering pipeline leveraging the vast amount of CAD-models obtained from [7] to generate a large synthetic training set. Moreover, they propose a geometry-aware classification loss to encourage correlation among neighboring views. Similarly, [10] uses a multi-task CNN pretrained by ImageNet based on AlexNet [8] or VGG16 [18] for joint detection and pose estimation. These methods predict the absolute pose rather than explicitly considering relations among different views of an object. In contrast, this paper aims to estimate the pose *difference* between views by taking into account the relative relations between any pair of views during learning.

To resolve ambiguities resulting from symmetries, [9] propose to incorporate geometric graph-matching constraints over the keypoint-features. Similarly, [20] combines the merits of global and local representations by jointly learning keypoint prediction and viewpoint estimation using CNNs. The method requires keypoint annotations. In contrast, we also aim to gain robustness to symmetry-confusion but without special annotation. We do so by faithfully embedding the topology of the 3D-rotation group, $SO(3)$, into our pose representation via hallucinating a given object’s views from other angles.

CAD-models have been used for pose estimation. [11] uses 3D-models and part-detectors to establish correspondences between CAD-models (of chairs) with real images. [14] learns to align parts detected by DPM at various viewpoints to a corresponding CAD-model. Similarly, [9] relies on CAD-models to align parts globally as well as locally in order to improve the pose alignment of objects. [19] also uses CAD-models but they render a large amount of 2D-images to learn robust pose classifiers. Similar to the reference, we rely on CAD-data to generate 2D-views for its precise viewpoint annotations and ease of generating sequences of rotating objects. Unlike [11, 9, 14], we do not employ CAD-models for any kind of point or part alignments, but rather use the rendered 2D-images from CAD-models to learn a pose representation describing the orbit of all poses.

The proposed method generates the orbit of all poses given a single view of an object and then measures the relative distance in azimuth by gauging the orbit of this object against another object’s orbit using orbit metric as illustrated in Figure 2.

3 Background

Definition 3.1. (Group) A group is a tuple (G, \cdot) consisting of a set G and its binary operation, $\cdot : G \times G \rightarrow G$, $(g, h) \mapsto g \cdot h$, where $g, h \in G$, satisfying axioms of *associativity*, *closure*, and the existence of an *identity* and an *inverse* element. We use G to denote a group for convenience.

We deal with a subgroup, G , of $SO(3)$ for azimuthal rotations around z -axis. The (sub-)group structure is given as: $G = \{e, g \mid e = g_K, g_k = \prod_1^k g\}$ where e, g and K denote the identity element, the generator and the order of the group, $|G|$, respectively. Group element g_k corresponds to k - rotations by applying generator g , k times. G is characterized by its generator matrix, $g \in \mathbb{R}^{3 \times 3}$, which acts on \mathbb{R}^3 . This subgroup is a *cyclic* group as applying g

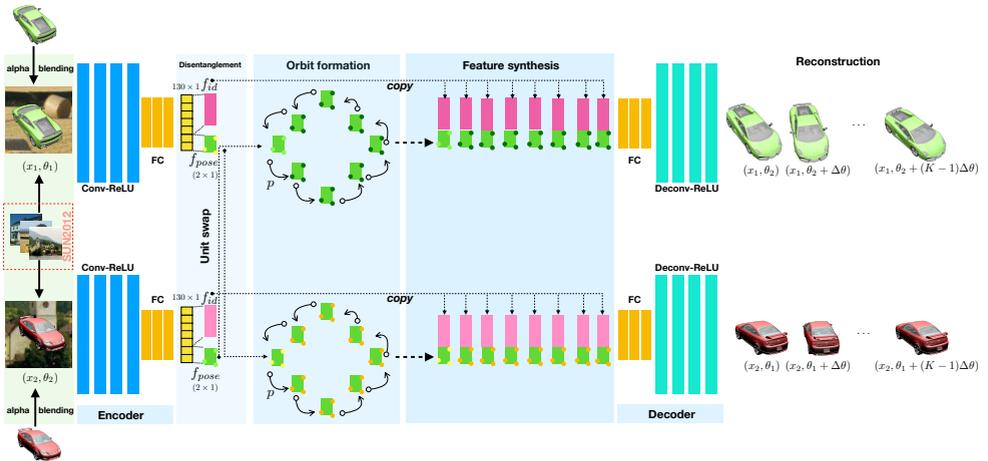


Figure 3: The proposed two-branch encoder-decoder architecture with shared parameters to learn the orbit generator. After learning, the orbit generator is simply the single branch without the decoder, which takes an input image and generates the full orbit in the latent pose space.

$|G|$ times brings the object back to its initial state. For rotations around the z -axis, g , is:

$$g = R_z(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad (1)$$

where $\theta \in [0, 2\pi)$ is the degree of rotation.

Definition 3.2. (Orbit) Let us denote the set, upon which group (G, \cdot) acts, with X . Then, an orbit of $x \in X$ undergoing a group transformation, G , is the subset of X such that $G \cdot x := \{g \cdot x \mid g \in G\}$.

Informally, the orbit is the set of measurements of an object, x , undergoing a group transformation, G . In this work, we evenly sample the azimuth into $K = 36$ discrete poses such that $\Theta = \{k \cdot \Delta\theta \mid \Delta\theta = \frac{2\pi}{K}, k = 0, \dots, K-1\}$. K determines $|G|$ and therefore the generator matrix is parametrized by $\Delta\theta$ such that $g = R_z(2\pi/K)$. In this way, an orbit is an ordered set of K samples on K consecutive poses.

4 Method

4.1 Orbit Generator

From a given view, the orbit generator should be capable of (i) inferring the initial pose of the input and (ii) constructing the rest of the orbit in latent pose space. We propose a two-branch encoder-decoder architecture with shared parameters to learn the orbit generator (see Figure 3). To ensure robustness against symmetries, we disentangle two intertwined factors, appearance and 3D-pose, on the final fully-connected layer of the encoder.

The *encoder* subnetwork receives a 64×64 RGB input image and projects it to the latent representation f_e with the map $\mathcal{F}_{encoder} : \mathbb{X}^{64 \times 64 \times 3} \rightarrow \mathbb{R}^{130}$. It consists of four convolutional layers and three fully connected layers all but the last one followed by a *ReLU*. For the final fully connected layer we opt for *tanh* to make the pose representation cover the range between $-c$ and c symmetrically. As the *ReLU* discards the negative half-plane it is unsuited here. All convolutional layers have a stride of 2 and a receptive field size of 5×5 .

For the *disentanglement* we construct a latent pose representation which varies only when the input is transformed by the subgroup G . Therefore, we use the irreducible representation to separate the appearance information from the pose information. The first 128 dimensions of f_e are designated to appearance independent of pose and the final 2 dimensions are designated to pose only. From now on, we refer to the former as $f_{id} (\in \mathbb{R}^{128})$ and to the later as $f_{pose} (\in \mathbb{R}^2)$.

Our goal is to establish a *homomorphism* between the subgroup $G \subset SO(3)$ defined in \mathbb{R}^3 and a group on the latent pose subspace, $(P, *)$. The homomorphism between G and P implies that f_{pose} will transform in an analogous manner as the action of $G \subset SO(3)$ in \mathbb{R}^3 . Hence, we have obtained an equivariant representation for the pose. The structure of the group P is similar to that of G , given as $P = \{e, p \mid e = p_K, p_k = \prod_1^k p\}$. We assume a matrix multiplication for the group operator, $*$. We adopt the analytical form of rotation matrix defined in \mathbb{R}^2 for its generator p such that the irreducible representation is a block-diagonal matrix of the form

$$\mathbf{R}(\Delta\theta) = \begin{bmatrix} 1 & & & \\ & \ddots & & \\ & & 1 & \\ & & & p \end{bmatrix} = \begin{bmatrix} \mathbf{1} & & & \\ & \cos(\Delta\theta) & -\sin(\Delta\theta) & \\ & \sin(\Delta\theta) & \cos(\Delta\theta) & \\ & & & \end{bmatrix} \quad (2)$$

where $\mathbf{1}$ is an identity matrix of size 128×128 . Applying $\mathbf{R}(\Delta\theta)$ consecutively leaves f_{id} invariant to rotations due to the identity block but transforms f_{pose} . In order to decrease computational redundancy due to large matrix-vector multiplications, i.e. $\mathbf{1}f_{id}$, we use vector slicing to separate f_{id} and f_{pose} and then only apply p on f_{pose} which implies only a $(2 \times 2) \times (2 \times 1)$ matrix-vector multiplication.

Due to the *projection* from 3D to 2D during image capture, the group structure which is observable in 3D is broken. Thus, we cannot establish a homomorphism relying on 2D-images. We impose three constraints on f_{pose} which are derived from the relationships of the group elements in order to get around this problem. Given two inputs (x_1, θ_1) and (x_2, θ_2) , the encoders compute the corresponding latent pose representations $f_{pose}(\theta_1)$ and $f_{pose}(\theta_2)$. We simultaneously impose three constraints on these latent representations in order to obtain orbits that satisfy the equivalence relations. The first constraint demands that the orbit is a circle over which each pose representation rests, as the group we are dealing with is a cyclic group. We denote it with \mathcal{L}_{radius} . The second constraint is derived from the definition of group and orbit. The constraint states that there must be a group action, p_* , that relates any given two elements of the orbit such that $f_{pose}(\theta_2) = p_* * f_{pose}(\theta_1)$. We refer to it as \mathcal{L}_{pair} . And the final constraint is a consequence of the symmetry-relation that orbits must satisfy: given $x, y \in X, y \in G \cdot x \iff x \in G \cdot y$. It implies that x is an element of y 's orbit if and only if y is in x 's orbit. It yields that any two f_{pose} units can be swapped as they belong to the same orbit. In Table 1, we provide the loss terms for each of the constraints to be used in the optimization.

After the pose swap, *local linear transformations* are applied to compute consecutive

| Constraint type | Loss term |
|-----------------|--|
| Radius | $\mathcal{L}_{radius} = c - \ f_{pose}\ _2 $ |
| Pair | $\mathcal{L}_{pair} = \ f_{pose}(\theta_2) - (\prod_1^N p) f_{pose}(\theta_1)\ _2, N = \frac{\theta_2 - \theta_1 \pmod{2\pi}}{\Delta\theta}$ |
| Symmetry | Unit swap |

Table 1: We impose three constraints on the latent pose representation, f_{pose} . Two of them are implemented into the objective function while the last one is implemented into the network architecture. We use $c = 0.8$ in all experiments.

pose unit corresponding to the next pose parameter. The next element on the circle can be computed from the previous one by matrix multiplication $f_{pose}(\theta' + \Delta\theta) = p * f_{pose}(\theta')$. We represent the orbit generated by the group P with the set, \mathcal{X}_p , ordered with respect to the rotations imposed by group elements: $\mathcal{X}_p = [e * f_{pose}, p_1 * f_{pose}, \dots, p_{K-1} * f_{pose}] \subset \mathbb{R}^2$.

The architecture of the *decoder* is mirrored from the encoder where the convolutions are replaced with deconvolutions. The identity unit is copied and concatenated with each and every orbit element in \mathcal{X}_p to give $f = [f_{id} \oplus (e * f_{pose}), \dots, f_{id} \oplus (p_{K-1} * f_{pose})]$. The decoder receives these feature representations and projects them back to the pixel space for reconstruction with $\mathcal{F}_{decoder}$. We denote the output of decoder block with $X_{decoder} = \mathcal{F}_{decoder}[f]$. Then we use the reconstructed image sequences $X_{decoder}^i$ and $X_{decoder}^j$ on both branches given a pair of input images (x_i, θ_i) and (x_j, θ_j) to optimize the model with

$$\mathcal{L}_{recon.} = \frac{1}{2KN_b} \sum_{i,j=1}^{N_b} \|X_{decoder}^i - X_{gt}^i\|_2^2 + \|X_{decoder}^j - X_{gt}^j\|_2^2$$

where N_b , K and X_{gt}^i are the batch size, the order of the group P and groundtruth images depicting the true sequence of the rotations for input image x_i .

After learning, the orbit generator is simply the single branch excluding the decoder. It takes an image as input and generates the full orbit in the latent pose space. The orbit in the latent space preserves all information between any views in the original observational space. The orbit representation is compact, a matrix of $K \times 2$ in our case. And, the orbit can easily be visualized to demonstrate where confusion in nearby poses and symmetries (opposite poses) originate.

4.2 Orbit Metric

To establish whether two poses are equal, one usually measures one pose, θ_1 , then the other one, θ_2 , and compares the two: $\theta_1 - \theta_2 \pmod{2\pi}$. We argue that a more robust difference between two poses is obtained from comparison of the two complete orbits.

An orbit metric between two orbits, \mathcal{X}_p^1 and \mathcal{X}_p^2 , is defined as:

$$\mathcal{M}_{\mathcal{X}_p^1 \leftarrow \mathcal{X}_p^2}(\delta) = \sum_{k=0}^{K-1} \langle \mathcal{X}_p^{(1,k)}, p_\delta * \mathcal{X}_p^{(2,k)} \rangle \quad (3)$$

$$\mathcal{M}_{\mathcal{X}_p^1 \leftarrow \mathcal{X}_p^2}(-\delta) = \sum_{k=0}^{K-1} \langle \mathcal{X}_p^{(1,k)}, p_\delta^{-1} * \mathcal{X}_p^{(2,k)} \rangle \quad (4)$$

$$\Delta\delta = \min \left\{ \underset{\delta=0, \dots, K-1}{\operatorname{argmax}} \mathcal{M}(\delta), \underset{\delta=0, \dots, K-1}{\operatorname{argmax}} \mathcal{M}(-\delta) \right\} \quad (5)$$

Subscript $\mathcal{X}_p^1 \leftarrow \mathcal{X}_p^2$ denotes that \mathcal{X}_p^1 is the reference orbit whereas \mathcal{X}_p^2 is the test orbit. We use the notation of $\mathcal{X}_p^{(i,k)}$ to denote the k th element in \mathcal{X}_p^i . Also, the second term inside $\langle \cdot, \cdot \rangle$ corresponds to a cyclic shift in \mathcal{X}_p^2 implying that all the elements are rotated by $\delta\Delta\theta$ or $-\delta\Delta\theta$ depending on the way orbits are compared.

Rather than the orbit metric (5), we prefer to use the directed orbit comparison, \mathcal{M} :

$$\Delta\delta = \operatorname{argmax}_{\delta=0,\dots,K-1} \mathcal{M}(\delta) \quad (6)$$

Ideally, the directed orbit comparison produces a sinusoidal signal varying with the shift parameter, δ . We estimate the pose of the test image by:

$$\theta_{est} = \theta_{ref} + \Delta\delta\Delta\theta \pmod{2\pi} \quad (7)$$

The directed orbit comparison satisfies the *non-negativity*, *identity of indiscernibles* and *triangular inequality* properties of a metric but fails the *symmetry* property. The directed comparison enables us to encode the direction to the left or to the right of the relative pose.

4.3 Training

In the first stage of learning the orbit generator, we need precise orientation data in order to arrive at precise orientation estimates. In practice, for cars (and many other objects) this implies a choice for synthetic training data as only synthetic data can guarantee accurate orientation annotations. For this reason, we acquire 1350 car models from ShapeNet [2] and render each one at 10° intervals of azimuth and elevation within $[-10^\circ, 50^\circ]$.

In the second stage, we infuse real data into the synthetic data to compensate for the lack of reality in the synthetic data. Since established benchmark datasets such as PASCAL3D+ do not contain images of the same object in different poses, we use real data only for imposing latent space constraints. We use the RMSProp optimizer with an initial learning rate of 10^{-4} which decays exponentially with $\gamma = 0.95$ every 10k iterations.

For data preprocessing, we use alpha-blending to embed rendered images onto randomly cropped backgrounds from the SUN2012 dataset [22] on the fly. Alpha blending is only applied to the input images excluding the target images used in reconstruction. For real images, we resize the detection boxes such that the longest dimension is 64 pixels and zero pad the remaining parts.

The *optimization objective* combines latent space constraints with the mean-squared reconstruction error :

$$\mathcal{L}_{total} = \beta_1 \mathcal{L}_{recon} + \beta_2 \mathcal{L}_{radius} + \beta_3 \mathcal{L}_{pair} \quad (8)$$

We take $(\beta_1, \beta_2, \beta_3) = (100, 1, 3)$ during the first stage of optimization and $(\beta_1, \beta_2, \beta_3) = (100, 1, 5)$ from then on.

4.4 Infer absolute pose from relative pose difference

Our method measures the relative pose difference between two views. Absolute pose of a view can be inferred by gauging it with another known pose. In this work, we simply use a synthetic image with its pose label θ_{ref} as the gauging example.

5 Experiments

5.1 Datasets

For evaluation, we use EPFL Car Dataset [12] and PASCAL3D+ [20].

EPFL Cars Dataset [12]. It contains varying-length sequences of 20 cars on a rotating stage. The dataset does not have exact labels for pose. Therefore, we follow the standard procedure [12] and produce approximate labels using the timestamps provided, assuming a constant angular velocity for the rotating platform. Following [2], we use first 10 cars for training and the remaining 10 for testing.

PASCAL3D+ [20]. This dataset is widely used to evaluate object detection and pose estimation tasks. It contains 12 rigid categories from PASCAL VOC 2012 [9] with 3D annotations. We use the train sets of PASCAL3D+ and ImageNet cars to train our model and use the PASCAL3D+ validation set for testing using RCNN detections¹.

Following standard protocol, the performance on EPFL is measured using *accuracy-36* (36 bins) [2] while the performance on PASCAL3D+ is measured using *AVP-24* (24 bins) [20].

5.2 Results

Our method is designed for measuring relative pose difference. However, in order to compare with state-of-the-art in pose estimation, here we evaluate on the basis of absolute pose.

| Method | accuracy-36 | Method | AVP-24 | |
|----------------|-------------|---------|-----------------------------|-------|
| 3D2PM-C [13] | 52.1% | DPM | [13] | 13.7% |
| 3D2PM-D [13] | 45.8% | | DPM-VOC+VP [13] | 24.6% |
| Fisher+spm [2] | 51.8% | Shallow | [2] | 15.9% |
| Decaf [2] | 45.9% | | Render For CNN [19] | 25.5% |
| This paper | 54.0% | | This paper | 28.3% |
| | | Deep | Viewpoints & Keypoints [20] | 40.0% |
| | | | Crafting MT-CNN [10] | 44.2% |

Table 2: Comparison with the state-of-the-art methods for pose recognition on EPFL (left) and PASCAL3D+ cars (right).

The results are summarized in Table 2. On EPFL, our method achieves state-of-the-art result, surpassing DPM-based [13] and [2] which relies on global CNN activations and separate classifiers. On PASCAL3D+ cars, our method compares favorably against DPM-based methods [14, 21] and deep learning methods [7, 19] with similar network capacity to ours.

Error modes. In the light of the study of [17], we investigate the error modes of our approach and compare against [14, 15, 20, 21]. The study classifies pose errors into three types: *nearby* ($15^\circ \leq \text{err} \leq 30^\circ$), *opposite* ($\text{err} > 165^\circ$) and *others* ($30^\circ < \text{err} \leq 165^\circ$). We present the results in Table 3. Our method achieves the lowest nearby-view error rate. In *opposite* case, [20] works the best, benefiting from the joint learning with keypoint estimations. Our method does not use keypoints as it requires additional keypoint annotations for learning, and still achieves a relatively low error rate in *opposite* case, better than DPM and Hough Forest based approaches. To conclude, the proposed method which takes into account

¹We use the RCNN detections provided by [10].

all the relative relations between any pair of views is advantageous in handling nearby and opposite poses.

| Method | nearby ($15^\circ < \text{err} < 30^\circ$) | opposite ($\text{err} > 165^\circ$) | others |
|-----------------------------|---|---------------------------------------|--------|
| VDPM [27] | 13.7% | 16.55% | 30% |
| DPM-VOC+VP[14] | 13% | 12% | 20% |
| BHF [15] | 14% | 11% | 54% |
| Viewpoints & Keypoints [20] | 12% | 5% | 18% |
| This paper | 9.6% | 8.9% | 35% |

Table 3: Comparison with the state-of-the-art in terms of error modes.

Orbits visualizations. Figure 4 visualizes the true orbits which is the collection of f_{pose} representations given each view. Dashed circle depicts the trace of a generated orbit starting from the filled initial view. Arrows indicate the ground-truth. Note that the arrows are not meant to show the absolute object orientation. They are used to illustrate the continuously changing pose of a car on a *anticlockwise* rotating stage. As shown in the figure, pose differences in the original observational space are properly captured in the orbits in the latent space thanks to equivariance. We can also see that some opposite views are confused, such as the two views in (b) highlighted by the red box.

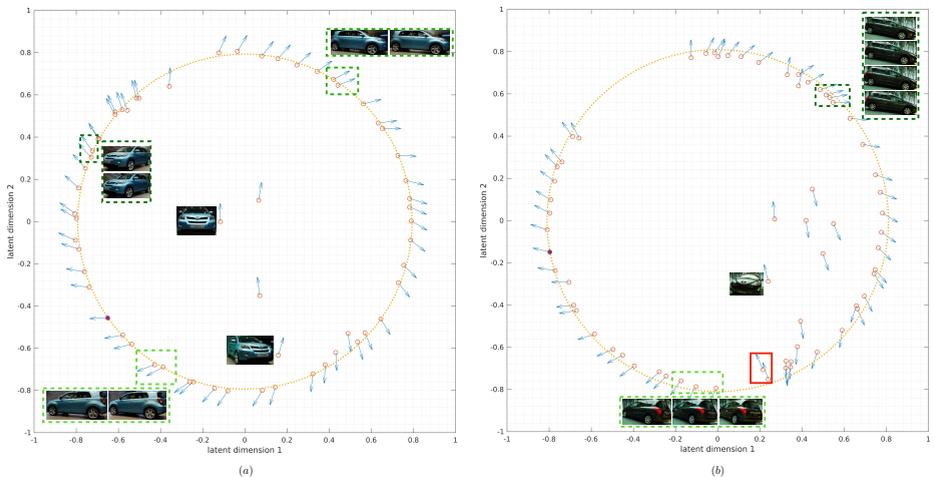


Figure 4: Visualizations of two orbits.

6 Conclusion

This paper proposes a new method for predicting pose difference by comparing orbits with a metric tailored for it. The method is capable of finding subtle differences in pose. Subtle differences in pose express intent which is important. We have demonstrated the effectiveness of the method on cars, where identifying small pose changes is hard due to their convexity but vital. For similar type of networks, we achieve the best performance.

References

- [1] Mathieu Aubry, Daniel Maturana, Alexei A Efros, Bryan C Russell, and Josef Sivic. Seeing 3d chairs: exemplar part-based 2d-3d alignment using a large dataset of cad models. In *CVPR*, 2014.
- [2] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical report, Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015.
- [3] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. *IJCV*, 2015.
- [4] Michele Fenzi and Jörn Ostermann. Embedding geometry in generative models for pose estimation of object categories. In *BMVC*, 2014.
- [5] Michele Fenzi, Laura Leal-Taixé, Bodo Rosenhahn, and Jörn Ostermann. Class generative models based on feature regression for pose estimation of object categories. 2013.
- [6] Michele Fenzi, Laura Leal-Taixé, Jörn Ostermann, and Tinne Tuytelaars. Continuous pose estimation with a spatial ensemble of fisher regressors. *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1035–1043, 2015.
- [7] Amir Ghodrati, Marco Pedersoli, and Tinne Tuytelaars. Is 2d information enough for viewpoint estimation? In *BMVC*, 2014.
- [8] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.
- [9] Joseph J. Lim, Hamed Pirsiavash, and Antonio Torralba. Parsing ikea objects: Fine pose estimation. In *ICCV*, 2013.
- [10] Francisco Massa, Renaud Marlet, and Mathieu Aubry. Crafting a multi-task cnn for viewpoint estimation. In *BMVC*, 2016.
- [11] James S. Milne. Group theory (v3.10), 2010. Available at www.jmilne.org/math/.
- [12] Mustafa Özuysal and Vincent Lepetit and Pascal Fua. Pose estimation for category specific multiview object localization. In *CVPR*, 2009.
- [13] Bojan Pepik, Peter V. Gehler, Michael Stark, and Bernt Schiele. 3d2pm - 3d deformable part models. In *ECCV*, 2012.
- [14] Bojan Pepik, Michael Stark, Peter V. Gehler, and Bernt Schiele. Teaching 3d geometry to deformable part models. In *CVPR*, 2012.
- [15] Carolina Redondo-Cabrera and Roberto Javier López-Sastre. Because better detections are still possible: Multi-aspect object detection with boosted hough forest. In *BMVC*, 2015.

- [16] Carolina Redondo-Cabrera, Roberto Javier López-Sastre, and Tinne Tuytelaars. All together now: Simultaneous detection and continuous pose estimation using a hough forest with probabilistic locally enhanced voting. In *BMVC*, 2014.
- [17] Carolina Redondo-Cabrera, Roberto Javier López-Sastre, Yu Xiang, Tinne Tuytelaars, and Silvio Savarese. Pose estimation errors, the ultimate diagnosis. In *ECCV*, 2016.
- [18] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, 2014.
- [19] Hao Su, Charles Ruizhongtai Qi, Yangyan Li, and Leonidas J. Guibas. Render for cnn: Viewpoint estimation in images using cnns trained with rendered 3d model views. In *ICCV*, 2015.
- [20] Shubham Tulsiani and Jitendra Malik. Viewpoints and keypoints. In *CVPR*, 2015.
- [21] Yu Xiang, Roozbeh Mottaghi, and Silvio Savarese. Beyond pascal: A benchmark for 3d object detection in the wild. In *WACV*, 2014.
- [22] Jianxiong Xiao, James Hays, Krista A. Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *CVPR*, 2010.