Learning 3-D Object Orientation from Images

Ashutosh Saxena **Justin Driemeyer** **Andrew Y. Ng**
Computer Science Department
Stanford University
Stanford, CA 94305
{asaxena,jdriemey,ang}cs.stanford.edu

Abstract

We propose a learning algorithm for estimating the 3-D orientation of objects. Orientation learning is a difficult problem because the space of orientations is non-Euclidean, and in some cases (such as quaternions) the representation is ambiguous, in that multiple representations exist for the same physical orientation. Learning is further complicated by the fact that most man-made objects exhibit symmetry, so that there are multiple "correct" orientations. In this paper, we propose a new representation for orientations—and a class of learning and inference algorithms using this representation—that allows us to learn orientations for symmetric or asymmetric objects as a function of a single image. We extensively evaluate our algorithm for learning orientations of objects from six categories.

1 Introduction

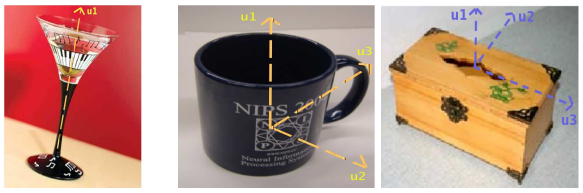
We consider the problem of learning the 3-D orientation of objects, such as a mug or a martini glass (Fig. 1), from an image. The ability to estimate orientation has many applications in vision and robotics. For example, the task of perception (object recognition) is considerably easier if an object's orientation is known. In other applications such as active vision/tracking, knowing the pose of a self-propelled object (e.g., an airplane) will also help to estimate its 3-D direction of motion. In robotics, estimating the pose of surrounding cars is also useful for autonomous driving, and knowledge of the 3-D orientation of an object is necessary to enable a robot to grasp it.

Unfortunately, the problem of estimating orientations is difficult because the space of orientations is non-Euclidean and non-linear. This property of orientations manifests in all methods of its representation [11]. In Section 3, we describe a number of representations for orientation, the problems associated with them, and explain why a naive attempt to learn using each of them would fare poorly. Symmetries in the object, which cause it to appear identical for multiple different orientations, cause further problems; in particular, the orientation is now ambiguous, in that there are multiple "correct" orientations. E.g., the box in Fig. 1c has identical faces, which makes it impossible to tell which side is the front. In Section 4, we describe a representation that addresses this problem.

In most prior work on estimating orientation, the orientations were assumed to lie within a small range of angles. In these cases, the problem is significantly easier because the orientations can be safely linearized (e.g., using Euler angles) around some "mean value," [19, 28] and discontinuities in the representation and ambiguities arising from symmetry do not have to be explicitly addressed.

In this paper, we propose a new representation, together with learning and inference algorithms, that allows us to estimate 3-D orientations as a function of features. Our representation and algorithms apply even in the presence of symmetries. We apply the algorithm to two tasks: (i) recognizing the pose of a new object (drawn from a known object class), and (ii) choosing at what orientation to orient a robotic arm/hand in order to grasp an object. In the latter case, the test object can be drawn from a *previously-unknown* object class.

2 Related Work



(a) (b) (c)

Figure 1: Images of some objects in different orientations.

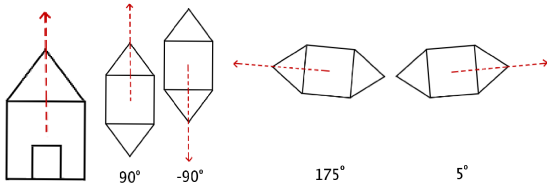
There is a large body of related work on "circular statistics," which deals with the cyclic nature of such data. For example, the Matrix-Fischer distribution [8, 13, 22, 11] is a Gaussian model restricted to a manifold. This literature considers a variety of representations for a fixed probability distribution over orientations y, but not the learning problem of estimating the conditional distribution of an orientation y given a set of features x; i.e., of estimating an orientation y as a *function* of x. (Because of the non-linearities and symmetries in the space of orientations and the discontinuities in the representations, they cannot be directly used with most standard learning algorithms.) One exception is the work on Spherical regression [5, 9, 29]; however this addresses a very special case—regressing orientations y against other orientations x. So if $y \in [-\pi, \pi)$ is an angle, then Spherical regression can use only a single feature $x \in [-\pi, \pi)$ (and similarly if y is a quarternion, then x must be also). None of this prior work considers symmetries, and neither have these ideas be developed for images.

Most work on learning on such non-Euclidean manifolds has focused on unsupervised learning of manifolds that are isometric to a subset of Euclidean space, e.g., [24, 30]. For non-isometric cases, algorithms such as locally smooth manifold learning [7] can be applied. [34] applied semi-supervised learning to multi-class classification. Notably, [20] gives an elegant method for supervised learning on non-linear manifolds such as a torus, using kernels with Laplacian eigenmaps. However, these methods address the problem of predicting a (discrete- or real-valued) target y, as a function of features x that lie on a non-linear manifold, such as a sphere or a torus. In contrast, our work addresses a different problem of predicting/regressing labels y that lie on a non-linear manifold (which can also be non-isometric). More recently, [16] have modeled data lying on multiple continuous manifolds, e.g., a torus, and have applied it to track people from a video sequence. Finally, multivalued regression [1] can be used to model cases when the output is multi-modal; however, it does not apply directly to the non-linearities and the ambiguities in the target space.

In vision, there are a few approaches that apply when one considers only 1-D orientations; for example, [4] gives a method based on analyzing the Fourier spectrum of simple line drawings. Multi-view object detection [31] is an approach in which objects are recognized at a few canonical poses. One can envision running an object detector at different poses to estimate 3-D orientation; however, this is not only a very indirect way to estimate orientations, but it would also be directly affected by the accuracy of the classifier, and would not apply to novel objects and objects from novel classes (as in our robotics application in Section 6.2). To our knowledge, work done in this literature, e.g., [31], presents results only on object detection, but not on general 3-D orientation estimation, and typically uses datasets that include views taken only from a circle around the object.

3 Representing Orientations

Here we describe a few popular representations for orientations, and explain why previous methods fail to learn them when the orientations are not clustered closely around a "mean value."



(a) (b)

Figure 2: (a) A hut has a unique orientation with $\theta \in [-180^o, 180^o)$. (b) A shape with 180^o symmetry. Red arrow (not part of the object) indicates an example of orientation $\theta = 90^o$, which is identical to $\theta = -90^o$. Restricting θ to lie in a half-space, such as $\theta \in [0, 180)$ does not help—this makes $\theta = 175^\circ$ and $\theta = 5^\circ$ very distant, even though they represent two nearly identical physical orientations.

Even in the case of planar 2-D objects, the representation suffers from problems. Consider the line drawing of a hut in Fig. 2a, which has a 1-D orientation that can be represented by $\theta \in \mathbb{R}$, with $\theta + 360^\circ = \theta$. Even if we restrict $\theta \in [-180^\circ, 180^\circ)$, this representation still suffers from a discontinuity at $\pm 180^\circ$. This makes naive learning of θ, such as with linear regression, impossible or

at best extremely difficult. For example, if in the training/test set distribution, the ground-truth orientations cluster around $\theta \approx 180^\circ$, then most objects' orientations will appear to be near either 180° or -180°. This makes the learning difficult since $E(\theta) = 0$, instead of being $+180^\circ$ or -180°. (E.g., 160° is closer to -160° than to 90°.) Wrapped Normal distribution [11] addresses this problem by modeling the circular nature of θ as $P(\theta|x, k; w) = \frac{1}{Z} \exp\left(-(\theta - w^T x - 2\pi k)^2/2\sigma^2\right)$. Here, $k \in I$ is a latent random variable. This model can be trained with EM [27].

For 3-D objects, the orientation can be described by 3 Euler angles [23]. However, learning using this representation is difficult because not only do each of the Euler angles wrap around $\pm 360^\circ$, but further at specific orientations two of the angles become degenerate. This discontinuity is called Gimble Lock. Further, the 3 angles cannot simply be learned separately using the wrapped Normal model, as they are interdependent. In prior work, such as [19], authors learned orientations but restricted to a *small* range (between 15°-20°) for various applications, e.g., face pose [17]. For such small ranges of angles that avoid Gimble lock, the angles can be linearized around a "mean value"; however, this fails for larger ranges of angles.

A quaternion [23] ($q \in \mathbb{R}^4, ||q||_2 = 1$) can also be used to represent 3-D orientation. However, quaternions suffer from the problem of anti-podal symmetry. I.e., q and $-q$ represent the same rotation. This means that for two identical orientations, we may end up with completely opposite values for their quaternion representations; this makes them impossible to learn using straightforward learning algorithms.[1] Rotation matrices R, which are orthogonal matrices ($R^T R = R R^T = I$) with $\det(R) = 1$, have a non-ambiguous representation. However, since all the elements are interdependent due to the orthogonality restriction, learning them is hard. [10]

4 Symmetry Invariant Representation

Symmetries in the object, which cause it to appear identical for multiple different orientations, cause further problems. In particular, the orientation is now ambiguous, in that there are multiple "correct" orientations for each object. Consider the shape shown in Fig. 2b, which exhibits 180° rotational symmetry. For any orientation θ and $\theta + 180$, the object appears exactly the same. Thus, any representation of its orientation will have two *opposite* values that correspond to the same visual appearance. This problem is exacerbated in 3-D. Most objects found in household and office environments exhibit some kind of symmetry. (See [12, 32] for a detailed description of symmetries.)

We first present our representation M for orientation that deals with these issues. To learn M, the representation should satisfy several criterion. First, M should be *invariant to ambiguities arising from symmetries*. Second, the representation should be *continuous*, i.e., two orientations that are physically close should be close in the representation. Finally, the representation should be *unique*, in that all orientations that look identical should have the same value.[2] Given a representation u for the orientation for a 2-D (planar), 3-D or more generally an n-D object, we will define a new representation $M(u)$, which our learning algorithm will estimate as a function of image features.

In 2-D, the orientation of an object (e.g., Fig. 2) can be completely described by a unit 2-vector $u = [\sin\theta, \cos\theta] \in \mathbb{R}^2$, where θ is the angle of rotation. In 3-D, an orientation can be completely described by two unit 3-vectors. E.g., the orientation of a mug (Fig. 1b) can be described by a unit vector representing the up direction of the mug, and a second one in the direction of the handle. (The third direction is redundant and can be obtained as a cross product of the first two.) However, for an object such as a martini glass that has only one distinguishable direction (pointing towards the top of the glass; Fig 1a), only one unit 3-vector is required to describe its orientation.

[1] To remove antipodal symmetry, one can consider the quaternions $q = [q_x, q_y, q_z, q_w]$ lying in a half-space $q_w > 0$. However, this still does not solve the problem because similar orientations will still be far in the representation. (see Section 6 and Fig. 5).

[2] Our representation will actually satisfy a stronger set of criteria of [14]: (i) *Uniform stretch*, The mapping should carry implicitly information about the distances in the original space and scales linearly with the angle between two hyper planes $||\delta M(u)|| = c||\delta u||$ for $||u||$ = constant. (ii) *Polar separability*: $||M(u)||$ is constant and carries no information about orientation; and (iii)*Uniqueness*: $M(u)$ has only one value for a given orientation. These condition ensure that the representation is *non-distorting*, in that all orientations have isomorphic mappings (manifold tangent spaces) in the representation.

In the general case of n-D objects, an orientation can be completely described by $p \leq (n-1)$ orthogonal unit n-vectors, where p is the number of distinguishable directions of the object in n-dimensions. More formally, the object's orientation can be described by a matrix $U \in \mathbb{R}^{n \times p}$ with $U^T U = I$; here, U's columns are $u_i \in \mathbb{R}^n$. The space of all such matrices U is called the Stiefel I-manifold [10]. For $p = n-1$, we can get a complete matrix $\tilde{U}$ by appending an orthogonal column such that $\det(U) > 0$. In 3-D, $\tilde{U} = [u_1, u_2, u_3] \in \mathbb{R}^{3\times 3}$ is a rotation matrix.

4.1 Representation for n-D objects

In this section, we define our representation for the general case of symmetries in n dimensional objects. We will describe, as examples, specialized cases for the representations for 2-D and 3-D objects in Section 4.2 and 4.3. For this general case, our representation M will be a higher order tensor. Space constraints preclude a lengthy discussion on tensors and group theory, but interested readers may refer to tutorial texts such as [18] and [33].

Below, we will let $\chi \in \mathbb{R}^{m\times 1}$ be a representation of an orientation in n-D. Some representations are non-ambiguous. E.g., if $R \in \mathbb{R}^{n\times n}$ is a rotational matrix ($RR^T = R^T R = I$, $\det(R) = 1$) representing an orientation in n dimensions, we can define χ to be the vectorized form of R, i.e., $\chi = R(:) \in \mathbb{R}^{n^2\times 1}$. Alternatively, some representations can be ambiguous, e.g., a quaternion for 3-D orientations in which q and $-q$ represent the same orientation. (This is called a double cover of SO(3).) Therefore, for $\chi = q$, we also have that $\chi' = -q$ represents the same orientation. In general, for a point χ representing an orientation in n-D, there could be other points χ' that represent the same orientation. The set of all points that represent the same orientation as χ is called the *cut-loci* of χ.

For a given orientation χ, let $\psi(\chi)$ denote the set of all representations χ' that result in the object appearing identical to χ, either because of cut-loci (where multiple values of χ correspond to the same physical orientation) or because of symmetries (so that multiple physical orientations correspond to the same appearance). E.g., if χ is a quaternion and the object considered is asymmetric, then $\psi(\chi) = \{\chi, -\chi\}$. Similarly, if χ_θ represents the 1-D orientation θ of a planar object that exhibits 3-fold ($120°$) rotational symmetry, then $\psi(\chi_\theta) = \{\chi_\theta, \chi_{\theta+120}, \chi_{\theta+240}\}$. Now, we define our representation $M(\chi)$ that allows learning even in presence of cut-loci and symmetries as:

$$M(\chi) = -\sum_{\{\chi_1,\ldots\chi_c\}\in \text{Permutations}\{\psi(\chi)\}} \text{Tprod}(\chi_1, ..., \chi_c) \quad (1)$$

where, $\text{Tprod}(.)$ is the tensor (or outer) product of the vectors, and $c = \text{card}(\psi(\chi))$. The summation is over all permutations of cut-loci and symmetries; this ensures that $M(\chi)$ gives the same value for all different cut-loci and symmetries of χ, and still satisfies the criterion of [14] (see footnote 2).

Although Riemannian manifolds, in general, could have cut-loci with an uncountable number of points, orientations can always be represented with only a finite number of points in their cut-loci.[3] For example, we can represent n-D rotations as the special orthogonal group SO(n) with no cut-loci, and quaternions in 3-D with one point in the cut-loci. The special Euclidean group SE(n) which jointly represents location and orientation also has no cut-loci.

4.2 Representation for 2-D objects

All symmetries in 2-D that have the same appearance for different orientations can be expressed as a N-fold rotational symmetry, e.g., a hexagon has 6-fold rotational symmetry. We define our representation as $M_N(\theta) = [\cos N\theta, \sin N\theta]$, which has the same value for symmetric orientations. E.g., $M_N(\theta + 2\pi/N) = [\cos(N\theta + 360), \sin(N\theta + 360)] = M_N(\theta)$. In [26], authors used this representation, but it can be shown that this is a special case of the general form in the present paper. Specifically, for 2-fold rotational symmetry, we have $\chi_1 = [\cos\theta, \sin\theta]$ and $\chi_2 = [\cos(\theta + 180), \sin(\theta + 180)]$. Now, $M(\chi_1) = -\text{Tprod}(\chi_1, \chi_2) - \text{Tprod}(\chi_2, \chi_1) = [2\cos^2\theta, 2\cos\theta\sin\theta; 2\cos\theta\sin\theta, 2\sin^2\theta] = [1+\cos 2\theta, \sin 2\theta; \sin 2\theta, 1-\cos 2\theta]$. I.e., up to an additive constant, $M_N(\theta)$ is same as our $M(\chi)$.[4]

[3] For example, if we represent 1-D orientations directly as $\theta \in \mathbb{R}$, the cut-loci would be the points $\theta \pm n2\pi, n = 1, 2, 3....$ However, the representation $[\cos\theta, \sin\theta]$ has no cut-loci.

[4] Similarly, for N-fold symmetry, one can see that the tensor product would result in N^{th} order terms in $\cos\theta$ and $\sin\theta$, which after summing over permutations of symmetries result in $\cos N\theta$ and $\sin N\theta$ terms.

4.3 Representation for 3-D objects

Most objects belong to one of the symmetry classes shown in Fig. 3. We will describe, as examples, our representation $M(u_1, u_2, u_3)$ given in Section 4.1, specialized to each of these cases.

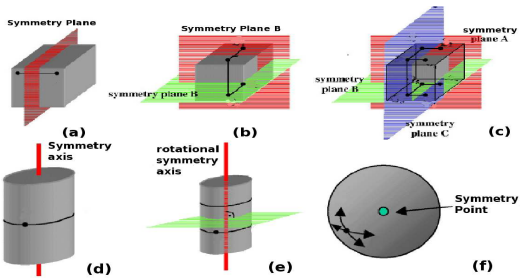


Figure 3: Various symmetries.

1. *No symmetry.* If the object is completely asymmetric, then $\{u_1, u_2, u_3\}$ completely specify the orientation of the object without ambiguity. Thus, we can safely choose our representation to be $M(u_1, u_2, u_3) = [u_1; u_2; u_3] \in \mathbb{R}^{9\times 1}$.

2. *Plane reflection symmetry.* Some objects exhibit plane reflection symmetry about one or more planes. For reflection around a single plane perpendicular to u_1 (Fig. 3a), we will have that u_1 and $-u_1$ are identical, while the other directions u_2 and u_3 will be non-ambiguous. We therefore define $M \in \mathbb{R}^{6\times 1} \times \mathbb{R}^{3\times 3}$ to be the tuple $([u_2; u_3], \ u_1 u_1^T)$. This representation[5] has the same value for the symmetric orientations u_1 and $-u_1$. Similarly, for dual plane reflection symmetry (Fig. 3b), we define $M = (u_3, \ [u_1; u_2][u_1; u_2]^T) \in \{\mathbb{R}^{3\times 1} \times \mathbb{R}^{6\times 6}\}$. For triple plane reflection symmetry (Fig. 3c), we define $M = [u_1; u_2; u_3][u_1; u_2; u_3]^T \in \mathbb{R}^{9\times 9}$. (Only the 3x3 block diagonal elements are more relevant in this case.)

3. *Rotational symmetry.* These symmetries exist when an object is symmetric about an axis, e.g., the rectangular box in Fig. 1c will appear the same after a 180° rotation. For 2-fold rotational symmetry in 3-D along the axis u_1, we define $M(u_1, u_2, u_3) = \{u_1, \ [u_2; u_3][u_2; u_3]^T\} \in \{\mathbb{R}^{3\times 1} \times \mathbb{R}^{6\times 6}\}$, which is invariant to this symmetry.

4. *Axial spherical symmetry.* Consider rotationally symmetric objects such as a martini glass (Fig. 1a) or a cylinder (Fig. 3e). We need only one vector u_1, lying along that axis, to fully describe its orientation. A martini glass has standard axial symmetry (the two directions u_1 and $-u_1$ are distinct); therefore we define $M(u_1) = u_1$. A cylinder has plane reflection axial symmetry (u_1 and $-u_1$ are identical); therefore we define $M(u_1) = u_1 u_1^T$.

5. *Spherical Symmetry.* Spherical symmetry (Fig. 3f) is trivially learned, and we define $M = 1$.

5 Learning

M gives a symmetry-invariant, continuous, and unique representation for orientation. In most cases, $M = y_1$ or $M = (y_1, y_2)$ where each y_i is either a vector or a rank-1 matrix. We will use a learning algorithm to separately estimate each of the components y_i as a function of image features x_i, and apply an inference algorithm to recover an orientation from this estimate.

5.1 Features

Standard image features such as in [31] would fare poorly for learning orientation, so we designed features better suited to the task. We start by partitioning the image into four quadrants and four radial segments (Fig. 4), by fitting an ellipse to the edge-image of the object. This gives a total of $4 * 4 = 16$ regions. For each region, our features will be the angles $\theta \in [0, 360)$ of the local edges.

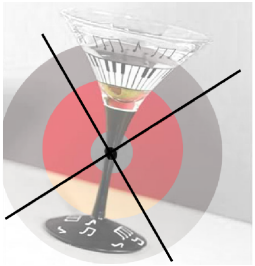


Figure 4: The feature vector for an object, showing the 4 quadrants, each having four radial segments.

However, the raw angles θ do not correlate well to the target values y that we need to predict. Therefore, we map our basic angles θ into the same form as the target y. For 3-D objects, y is made of a combination of circular functions of the form $\sin\alpha$ and $\cos\alpha$ for asymmetrical objects, and pairwise products $\sin\alpha\cos\alpha$, $\sin^2\alpha$, etc. for symmetrical objects. Therefore, our features will also be the corresponding circular

[5] M would be a grade 2 element in the Clifford algebra [21]. I.e., it consists of two parts: a vector in $\mathbb{R}^6$, and a matrix in $\mathbb{R}^{3\times 3}$.

functions of θ, and have the same dimension as y. For a given target angle, the edges are often distributed around that angle, e.g., a pencil at 40^0 will have edges between 30^0 and 50^0. Since $E[\sin(x)] \neq \sin(E[x])$, to capture the distribution of edges, we also use harmonics of the functions, i.e. $\sin(k\theta)$. Finally, we obtain our full feature vector X by concatenating the histogram of the features for each of these 16 regions. Note that each individual feature has the dimension as the target y, and so the overall feature vector X is a concatenation of many such features. So if $Y \in \mathbb{R}^m$, then $X \in \mathbb{R}^{m\times k}$; if $Y \in \mathbb{R}^{m\times m}$, then $X \in \mathbb{R}^{m\times m\times k}$ is a tensor.

5.2 Probabilistic Model

We estimate M as a function of the image features. Depending on the type of symmetry, M could be a vector, matrix, tensor or their combination; therefore, we will use the corresponding/appropriate form of the Gaussian distribution to model it. For $Y \in \mathbb{R}^m$ (the vector part) and $Y \in \mathbb{R}^{m\times m}$ (the matrix part),[6] we have:

$$P(Y|X;W,K) = |2\pi K^{-1}|^{-n/2} \exp\left[-\tfrac{1}{2}\text{Tr}(K(Y-XW)^T(Y-XW))\right] \quad (2)$$

Here, X are the features of the image, W are the parameters of the model, and K^{-1} is the shared covariance matrix. Note that this is a deficient model [3], since it allows positive probability even for invalid configurations. More formally, $y^T y = 1$ for the vector part, and $Y = uu^T$ is symmetric positive definite and of rank 1 for the matrix part. Choosing symmetric features allows us keep our estimate of M symmetric, but this model allows Y to take values where $\text{rank}(Y) \neq 1$. We learn the parameters W and K of the Gaussian model by maximizing the conditional log likelihood $\log \prod_i P(M_i|X_i;W,K)$ using the training set data.

Inference: Given an image with features X and a learned model with parameters K and W, we now describe an inference procedure for computing the MAP estimate of an object's orientation under the model. First, consider the case where $y \in \mathbb{R}^{m\times 1}$ is the vector part of M. Since $y^T y = 1$, our MAP estimate for y is given by $\arg\max_{y:y^Ty=1} \log P(y|X;W,K) = \arg\max_{y:y^Ty=1} TrKy^TXW$. The closed form solution of this is $y = XW/||XW||_2$.

Next, we consider the case where $Y = uu^T$ is the matrix part of M. Note that the conditions $Y \in \mathbb{R}^{m\times m}$ is positive definite, symmetric and of rank 1 are sufficient to ensure that Y is of the form $Y = uu^T$. For a new test image with features X, the MAP estimate for Y is:

$$\begin{aligned} \arg\min_{Y} \quad & -Tr(KYXW) \\ \text{s.t.} \quad & Tr(Y) = 1, Y \geq 0, \text{Rank}(Y) = 1 \end{aligned} \quad (3)$$

The optimization problem in Eq. 3 is non-convex. We solve it approximately by taking a semi-definite relaxation [6], thus dropping the rank constraint to obtain the convex optimization problem in Eq. 3. Finally, u is obtained by taking the eigenvector corresponding to highest eigenvalue of Y. To get the full rotation matrix, we first form the rotation matrix $\hat{R}$ by rearranging the inferred u and then project $\hat{R}$ into the orthogonal subspace as $R = \hat{R}(\hat{R}^T\hat{R})^{-1/2}$.[7]

6 Experiments

We trained our supervised learning algorithm using synthetic images of objects, and tested it on the tasks of inferring 3-D orientation from single images of different real objects from the object class.

To apply our supervised learning algorithm, we required a labeled training set, i.e., a set of images labeled with the 3-D orientation of the object. Since, collecting real data is cumbersome and manual labeling is prone to errors, we chose to learn from synthetic data generated using computer graphics that is automatically labeled with the correct orientations. In detail, we generated 9400 labeled examples comprising objects from six object classes, with random lighting conditions, camera position, object orientation, etc. We quantitatively evaluated the algorithm on real data. For this, we built

[6] For the general case, we would use the tensor form of the Gaussian model. [2]

[7] In our experiments, Y was almost always close to being rank 1, and $\hat{R}$ almost always close to being orthogonal. For the general case of other symmetries, following similar steps, we would first drop the rank constraint, infer the higher order tensor Y, and then perform a rank-one approximation to the tensor. [35, 15]

a custom setup to collect ground-truth labeled data using the markers while capturing the images. The algorithm was used to predict the 3-D orientation from these images (with the markers cropped out).

Definition of Error: We report errors in rotation angle—the angle by which the predicted orientation must be rotated to get to the actual orientation. In higher dimensions, however, this error can be quite non-intuitive. E.g., for an asymmetric object in 3-D, the mean error given by an algorithm predicting random orientations would be 120^0 (not 90^0). Presence of symmetries make this measure even more non-intuitive. Therefore, we define a more informative error metric, "Fraction-error", to be the fraction of orientations (sampled uniformly from all orientations) that are better than the prediction. (It is 0.0 for exactly correct, 0.5 for random, and 1.0 for maximally incorrect predictions.)

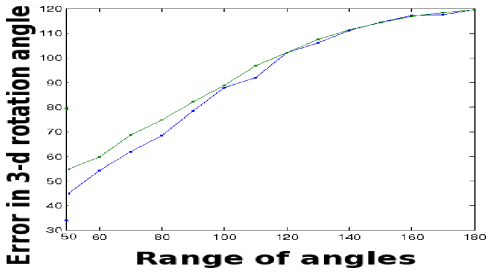


Figure 5: Test set error (blue) vs. range of angles. As the range of angles considered is increased, performance of the half-space quaternion method decreases rapidly. Error of a baseline that predicts the mean quaternion is shown in green.

6.1 Results on Objects

We provide extensive evaluation of our algorithm on a test set comprising real images of objects, from 6 classes: (i) Long cylindrical objects: pen, hex-driver, spoon, etc., (ii) Wine glasses: martini glasses, goblet shaped glass, etc., (iii) Mugs: different sizes/shapes, (iv) Tea cups: different shapes/sizes, (v) Boxes: white board erasers, wooden blocks, etc., (vi) Staplers: different examples.

We used 10-20 images of each of the 3-5 objects from each object class. (Some examples of the objects tested on are shown in Fig. 6.) In addition, we also test our algorithm on about 400 synthetic images for each object class. We perform comparisons of the following algorithms:
(a) *Wrapped Normal (1-D)*: Angles learned using the Wrapped Normal distribution.
(b) *Half-space quaternions*: Learn quaternions, restricted to a half-space $q_1 \geq 0$.
(c) *No features*: Learning our representation using our method, but without any image features. This effectively predicts the "mean" orientation, and therefore is a baseline for comparison.
(d) *Rotation matrices*: Here, we learn the rotation matrices directly by using linear regression, without considering symmetries in image features and in the object.
(e) *Our model with naive inference*: In this model, we show the results by directly taking $Y = XW$, i.e., without using the SDP inference method proposed.
(f) *Our full model*: Using our full algorithm.

We first show that the error increases significantly when the training and test sets contain a large range of orientations, and not only a small range of orientations clustered around some "mean value." Fig. 5 shows the 3-D rotation angle error as a function of the maximum angle away from the mean value, using half-space quaternion method. Table 1 shows that when we consider the full space of 3-D orientations, approaches that use the most straightforward representations of orientation, such as the rotation matrices, do not perform well. Table 1 presents the rotation angle error and fraction error for the different algorithms on a variety of objects. We report results on learning both 1-D orientation (for axially symmetric objects, where the task is to predict u_1 projected into the image plane) and 3-D orientation. In all cases, our algorithm significantly outperforms simpler methods.

When the axis of rotation is in the image plane, our algorithm cannot distinguish whether the compression (e.g., change in length of a pen) is due to rotation or due to the physical object being smaller. This is one of the major sources of error in our algorithm. Indeed, the errors in estimating the orientation projected into the image plane (corresponding to a rotation around the axis normal to the image plane) are usually quite low (e.g., 3.2° for long cylindrical objects).

Our algorithm appears to generalize very well. After being trained on synthetic images (from a known object class), it is able to predict orientations of objects belonging to new objects from the same class. For example, after being trained on pencils, it predicted well on a knife; and after being trained on martini glasses, it predicted well on wine glasses as well. Some object instances were quite different in shape than the synthetic examples trained on, e.g., the white tea cup in Fig. 6.

Mugs Wine Glasses Long objects Cups Boxes Staplers

Figure 6: Typical examples of predicted orientations (solid lines) for some real test objects, and their ground-truth orientations (dashed). (Best viewed in color.)

Table 1: Average absolute rotation error (fraction error) in predicting the orientation for different objects. Training on synthetic images of objects, and prediction on different test images.

TEST ON SYNTHETIC OBJECTS

TESTED ON	MUGS	WINE GLASS	LONG OBJECTS	TEA CUPS	BOXES	STAPLERS	ROBOTIC ARM
SYMMETRY	1-REFLECT	AXIAL	AXIAL, 1-REF	1-REFLECT	3-REFLECT	1-REFLECT	3-REFLECT
WRAPPED NORMAL (1-D)	-	24.1 (.25)	7.5° (.08)	-	-	-	-
OUR ALGORITHM (1-D)	-	**4.5° (.05)**	**2.6° (.03)**	-	-	-	-
ROTATION MATRICES (3-D)	74.3° (.74)	116.9° (.54)	68.8° (.65)	71.6° (.69)	69.9° (.67)	70.2° (.69)	66.2° (.64)
NO FEATURES (3-D)	48.7° (.45)	88.0° (.49)	51.7° (.44)	55.0° (.59)	44.4° (.42)	46.5° (.45)	46.4° (.45)
NAIVE INFERENCE (3-D)	42.3° (.45)	42.2° (.20)	18.1° (.20)	39.8° (.37)	24.5° (.24)	31.2° (.29)	38.0° (.35)
OUR ALGORITHM (3-D)	**18.4° (.17)**	**27.3° (.11)**	**11.9° (.04)**	**21.4° (.20)**	**12.8° (.11)**	**22.3° (.23)**	**22.2° (.20)**

TEST ON REAL OBJECTS

WRAPPED NORMAL (1-D)	-	28.9 (.29)	12.8° (.14)	-	-	-	-
OUR ALGORITHM (1-D)	-	**6.5° (.07)**	**3.2° (.04)**	-	-	-	-
ROTATION MATRICES (3-D)	66.9° (.62)	118.7° (.55)	66.0° (.62)	67.2° (.62)	71.7° (.70)	64.2° (.59)	58.0° (.51)
NO FEATURES (3-D)	49.4° (.45)	91° (.51)	54.1° (.46)	50.0° (.48)	54.0° (.59)	47.7° (.46)	48.0° (.45)
OUR ALGORITHM (3-D)	**26.8° (.24)**	**24.0° (.10)**	**16.7° (.06)**	**29.3° (.28)**	**13.1° (.14)**	**26.6° (.24)**	**26.0° (.23)**

6.2 Robotic Applications

We used this algorithm in the problem of grasping novel objects using robotic arms [25, 26]. Specifically, we are given an image of an object, which can be a previously-unseen object, from a previously-unknown object class. Our task is then to choose an orientation for the hand of our robot arm so as to enable the robot to correctly grasp the object. For example, given a picture of a long pencil lying on a table, we should choose an orientation in which the robot's two fingers are perpendicular to the pencil's main axis, rather than parallel to it. Typically 30° accuracy is needed to successfully grasp an object, which our algorithm almost always attains (Table 1, last column).

7 Conclusion

We presented an algorithm for learning 3-D orientation of objects from a single image. Orientation learning is a difficult problem because the space of orientations is non-Euclidean, and in some cases (such as quaternions) the representation is ambiguous, in that multiple representations exist for the same physical orientation. We presented a symmetry invariant, continuous, unique representation to address these problems, together with efficient learning and inference algorithms using this representation. We evaluated our algorithm on the task of estimating the 3-D orientation of new objects from six different object categories.

References

[1] A. Agarwal and B. Triggs. Monocular human motion capture with a mixture of regressors. In *IEEE Workshop Vision Human Computer Interaction, CVPR*, 2005.

[2] P. J. Basser and S. Pajevic. A normal distribution for tensor-valued random variables: Applications to diffusion tensor mri. *IEEE Tran Medical Imaging*, 2003.

[3] P. Brown, S. Pietra, V. Pietra, and R. Mercer. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19, 1993.

[4] J. Bugun, G. H. GranLund, and J. Wiklund. Multidimensional orientation estimation with applications to texture analysis and optical flow. *IEEE PAMI*, 13(8), 1991.

[5] T. Chang. Spherical regression and the statistics of tectonic plate reconstructions. *Int'l Stats Rev*, 1993.

[6] A. d'Aspremont, L. El Ghaoui, M. Jordan, and G. Lanckriet. A direct formulation for sparse pca using semidefinite programming. In *NIPS 17*, 2004.

[7] P. Dollar, V. Rabaud, and S. Belongie. Non-isometric manifold learning: Analysis and an algorithm. In *ICML*, 2007.

[8] T. D. Downs. Orientation statistics. *Biometrika*, 59:665, 1972.

[9] T. D. Downs. Spherical regression. *Biometrika*, 90:655, 2003.

[10] S. Fiori. Unsupervised neural learning on lie group. *Int'l Journal Neural Sys*, 12(3):219–246, 2002.

[11] N. Fisher. *Statistical Analysis of Circular Data*. Cambridge University Press, 1993.

[12] M. Kazhdan, B. Chazelle, D. Dobkin, T. Funkhouser, and S. Rusinkiewicz. A reflective symmetry descriptor for 3d models. In *Algorithmica*, 2003.

[13] C. Khatri and K. Mardia. The von mises-fisher matrix distribution in orientation statistics. *J. Royal Stat Soc*, page 95, 1977.

[14] H. Knutsson. Producing a continuous and distance preserving 5-d vector representation of 3-d orientation. In *Workshop Comp Arch Pattern Anal Images DB mgmt*, 1985.

[15] L. D. Lathauwer, B. D. Moor, and J. Vandewalle. A multilinear singular value decomposition. *SIAM J. Matrix Anal. Appl.*, 21, 2000.

[16] C.-S. Lee and A. Elgammal. Modeling view and posture manifolds for tracking. In *ICCV*, 2007.

[17] C. Malsburg. Tracking and learning graphs and pose on image sequences of faces. In *Auto Face Gesture Recog.*, 1996.

[18] P. McCullagh. *Tensor Methods in Statistics*. Monographs on Statistics and Applied Probability, 1987.

[19] P. Mittrapiyanuruk, G. N. DeSouza, and A. C. Kak. Calculating the 3d-pose of rigid objects using active appearance models. In *ICRA*, 2004.

[20] J. Nilsson, F. Sha, and M. I. Jordan. Regression on manifolds using kernel dimension reduction. In *ICML*, 2007.

[21] I. Porteous. *Clifford algebras and the classical groups*. Cambridge University Press, 1995.

[22] M. J. Prentice. Orientation statistics without parametric assumptions. *J.R. Stat Soc B*, 48, 1986.

[23] B. Rieger and L. J. Vliet. A systematic approach to nd orientation representation. *Image Vis Comp*, 2004.

[24] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323, 2000.

[25] A. Saxena, J. Driemeyer, J. Kearns, and A. Y. Ng. Robotic grasping of novel objects. In *NIPS*, 2006.

[26] A. Saxena, J. Driemeyer, and A. Y. Ng. Robotic grasping of novel objects using vision. *IJRR*, 27:157–173, 2008.

[27] F. H. Seitner and B. C. Lovell. Pedestrian tracking based on colour and spatial information. *Digital Image Computing: Techniques and Applications*, 2005.

[28] G. Shakhnarovich, P. Viola, and T. Darrell. Fast pose estimation with parameter sensitive hashing. In *ICCV*, 2003.

[29] M. Stephens. Vector correlation. *Biometrika*, 66, 1979.

[30] J. B. Tenenbaum, V. D. Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290:2319–2323, 2000.

[31] A. Thomas, V. Ferrrari, B. Leibe, T. Tuytelaars, B. Schiele, and L. Gool. Towards multi-view object class detection. In *CVPR*, 2006.

[32] S. Thrun and B. Wegbreit. Shape from symmetry. In *ICCV*, 2005.

[33] M. Tinkham. *Group Theory and Quantum Mechanics*. Dover Publications, 2003.

[34] H. Wang, S. Yan, T. Huang, J. Liu, and X. Tang. Transductive regression piloted by inter-manifold relations. In *ICML*, 2007.

[35] T. Zhang and G. H. Golub. Rank-one approximation to high order tensors. *SIAM J Mat Anal App*, 2001.