Seminar SS2018: Learning 3-D Orientation from Images

Tridivraj Bhattacharyya (30335538)
Supervisor: Max Schwarz

Rheinische Friedrich-Wilhelms-Universität Bonn

Abstract. This is a critical analysis of the paper "Learning 3-D Object Orientation from Images". [5] The problem of estimating orientation of objects has been tackled in many different ways previously. The primary challenge arises from orientations being in general non-linear and non-Euclidean. Most previous also do not deal well with symmetries. The paper proposes a model to represent orientation while considering ambiguities arising from symmetries. This model is used by an inference learning algorithm to predict an orientation of an object based on the features extracted from the image of the object.

1 Introduction

Orientation can be defined as the relative position or direction of an object. In the context of computer vision, this would be in which direction a specific part/feature of an object points to (e.g. The front end of a pencil). Mathematically orientation angles tend to be in non-Euclidean space and the modelling of angles itself is circular instead of linear. The following topics are explain briefly before summarizing the paper.

1.1 Relevant Topics

Quarternions Quaternions can be described as an extension of complex numbers which can also be used to represent 3-D rotations. They generally have the form $a + ib + jc + kd$ where $a, b, c, d \in \mathbb{R}$ and i, j, k are complex numbers. Therefore there are three imaginary axes and one real. This can be used to represent 3-D rotation, reflection and scaling. The primary use of quaternions in place of Euler angles is to avoid the Gimbal Lock problem that may arise.

Maximum A Posteriori Maximum a Posteriori is an estimate of some unknown quantity based on some prior distribution of data. This estimate is ideally equal to the one that is most frequent in the distribution. eg. After flipping a coin 3 times, the probability of getting heads in the 4-th try can be represented as,

$$h_{MAP} = \arg\max_h P(h|D) = \arg\max_h \frac{P(D|h)P(h)}{P(D)} = \arg\max_h P(D|h)P(h)$$

where, $P(h)$ = probability of getting heads,
$P(D)$ = probability distribution over 3 previous flips

In the above example, $P(D)$ does not depend on h. Therefore it can be ignored.

1.2 Previous Works

A significant amount of work has already been done is estimating orientation of objects. This includes working with both Euclidean and non-Euclidean spaces. The paper initially mentions modelling estimation based on circular statistics using the Matrix-Fischer distribution and Spherical Regression. However these methods were not proposed with the idea of images in mind.

The Locally Linear Embedding (LLE) algorithm by Roweis and Saul [4] works on estimating manifold structures by unrolling a higher dimensional manifold into a lower dimensional linear structure. However this works best for isometric structures where the distance is preserved even after conversion. The study of embedding in localized patches and its neighbourhood provides us with an idea about the shape of the underlying manifold. For non-isometric manifolds Dollar et al. [1] proposes an augmented version of the Locally Smooth Manifold Learning (LSML) to learn the structure of an object. Lee and Elgammal [2] propose a method to model continuous manifolds by taking images from different points along a view circle. They use this to track the 3-D configuration of a person continuously. Saxena et al. [6] use Markov Random Field to reconstruct 3-D models of unstructured environments from a single 2-D image. Thomas et al. [8] introduced an object classifier by combining the Implicit Shape Model with a multi-view object detector. Multiple images from random viewpoints are used for extracting features to classify the objects and recognize their pose. While the general problem being solved in each paper is similar, they apply different methods.

LLE and LSML both are similar to the current paper with regards to conditional orientation estimation based on feature extraction, but the process and modelling is different. The others while estimating orientation do not take into account the symmetry problem. Saxena et al [5] introduces a method which models orientation for both isometric and non-isometric manifolds while taking care of ambiguities arising from symmetries.

2 Summary

The primary contribution of the paper is providing a model to represent orientation that takes into account symmetries, if any. Symmetry means multiple instances in which the orientation of an object appears same. eg. A spherical ball being rotated around its centre or the two figures on the right hand side of

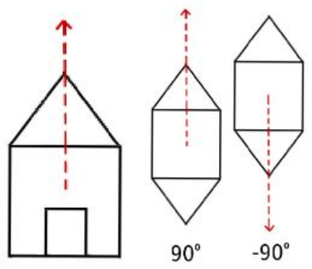

Fig. 1. 1-D Orientations [6]

Figure 1.

Naively, one can represent 1-D orientation in a linear manner. However the relationship of angular data is circular in nature. So by flattening it out discontinuities would rise at both ends of the line. eg. The angles 0° and 360° are actually the same position in a circle, whereas in a line they lie at the two opposite ends. An alternate is Wrapped Normal Distribution. It works by "wrapping" the normal distribution around an unit circle. Therefore it considers the circular property of orientation.

$$P(\theta|x, k; \omega) = \frac{1}{Z} exp(-\frac{\theta - \omega^{\intercal} x - 2\pi k)^2}{2\sigma^2}$$

where, θ = orientation angle,
σ = standard deviation of the unwrapped distribution.

In 3-D, the representations are a bit more complex. Ideally one would represent 3-D orientation with Euler angles. Basically these are the angles made by a rigid body with respect to each of the 3-D axes. While being the simplest way to represent orientation in 3-D, it suffers from a problem known as Gimbal Lock. If each of the axes of rotation are imagined as a gimbal, it is possible that two of the gimbals might line up. This would cause the body to be stuck into rotation of degenerate 2-D space by losing a degree of freedom. This can be visualized from Figure 3 [Wikipedia] where two of the axes are parallel. This can be especially problematic in Robotics and is known as "wrist flip" in Robot arms.

The problem of Gimbal Lock can be solved by representation in Quarternions (Section 1.1). However here we have the problem called "anti-podal" symmetry. This causes multiple identical orientation representations to have ambiguous values. eg. Consider an object with orientations q and $-q$ using Quarternions. These can be identical but are still opposite of each other. One can further use rotation matrices for 3-D orientation representation. They are non-ambiguous but each of their elements are interdependent based on orthogonality restriction. Therefore the idea is to create a model which is flexible for both asymmetric and symmetric objects.

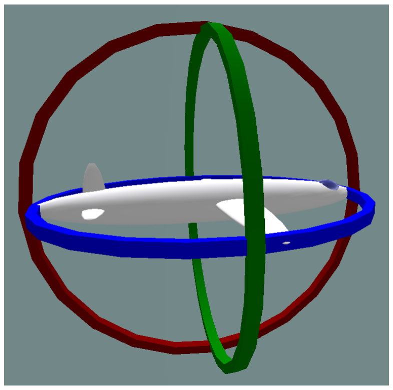

Fig. 2. 3-D Orientation without Gimbal-lock [Wikipedia]

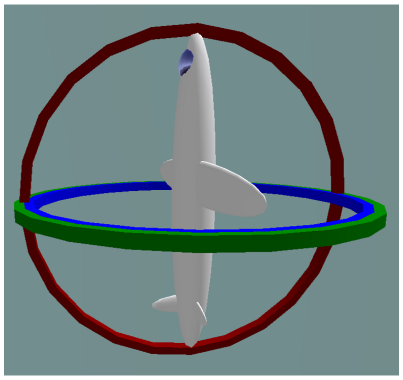

Fig. 3. 3-D orientation with Gimbal-lock [Wikipedia]

2.1 Model Representation

Let us consider the model to be $M(u)$ where u is the orientation representation in n-D. There are three condition that the model must abide by.

1. M must not be affected by symmetries
2. M must be continuous i.e. orientations which look similar have to lie close in M
3. All orientations must be unique in M

In n-D, this can be represented by a matrix $U \in \mathbb{R}^{n \times p}$ where, $p \leq (n-1)$ is the number of directions of the object in which an orientation can be considered. eg. For a coffee mug this would be the directions in which the top and the handle point to. The matrix must satisfy $U^{\intercal}U = 1$. In 3-D, this becomes a 3×3 rotation matrix. Moreover 3-D rotation can be represented by two or one unit-3 vectors depending on how many distinguishable directions of orientation the object possesses (two for the mug as mentioned). For 2-D representation, it is even simpler with a 2-D vector $u = [\sin\theta, \cos\theta]$ where θ is the angle of rotation.

While representing objects without ambiguities, one can use this model by flattening the rotational matrix $R \in \mathbb{R}^{n\times n}$ into a $\mathbb{R}^{n^2\times 1}$ vector. To represent symmetries, the identical orientations have to be collated as one first. This is done by considering $\psi(\chi)$ to be the set of all identical orientations with respect to the orientation χ. The model can then represent these ambiguities as,

$$M(\chi) = -\sum_{\{\chi_1,\cdots,\chi_c\}} T_{prod}(\chi_1,\cdots,\chi_c) \tag{1}$$

where, $\{\chi_1,\cdots,\chi_c\} \in Permuations\{\psi(\chi)\}$
$T_{prod}(.)$ = outer product of the vectors χ_i $[i \in \{1,\cdots,c\}]$
c = number of identical orientations to χ

In 2-D, this type of symmetry is termed as N-fold rotational symmetry for an object displaying symmetry N times. eg. The angle that each side of a hexagon makes with its centre is 60°. By rotating the hexagon by 60°, the object looks exactly the same. Since this can be done 6 times till it reaches its original configuration, this is called 6-fold rotational symmetry.

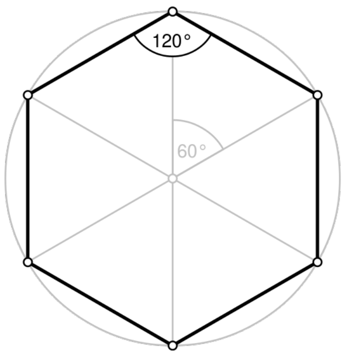

Fig. 4. Hexagon with 6-fold rotational symmetry [Wikipedia]

This can be represented as, $M_N(\theta) = [\cos N\theta, \sin N\theta]$. Using equation 1 for a 2-fold rotational symmetry,

$$M(\chi_1) = -T_{prod}(\chi_1,\chi_2) - T_{prod}(\chi_2,\chi_1) = \begin{bmatrix} 1+\cos 2\theta & \sin 2\theta \\ \sin 2\theta & 1-\cos 2\theta \end{bmatrix}$$

where, $\chi_1 = [\cos\theta, \sin\theta]$ and $\chi_2 = [\cos(\theta+180), \sin(\theta+180)]$ represent the identical orientations. The derivation is skipped here but can be found in the paper. The terms 2θ here represent the 2-fold symmetry exhibited. Intuitively one would get a bigger matrix with $N\theta$ terms for a N-fold symmetry.

In case of 3-D, there are in general 6 cases of symmetry to be considered. Let us consider the axes of rotation as u_1, u_2 and u_3. The following representations are modelled with the three rules as mentioned above.

1. Asymmetry - As mentioned before, the matrix can be flattened to get a vector. $M(u_1, u_2, u_3) = [u_1; u_2; u_3] \in \mathbb{R}^{9\times 1}$
2. Plane Reflection Symmetry - This happens when the the orientations are symmetric along one or more planes.
   - Single Plane - For ambiguity along a single plane u_1, $M(u_1, u_2, u_3) = ([u_2; u_3], u_1 u_1^\intercal \in \{\mathbb{R}^{6\times 1} \times \mathbb{R}^{3\times 3}\}$ (Figure 6a)
   - Double Plane - For ambiguity along the planes u_1 and u_2, $M(u_1, u_2, u_3) = ([u_3], [u_1; u_2][u_1; u_2]^\intercal \in \{\mathbb{R}^{3\times 1} \times \mathbb{R}^{6\times 6}\}$ (Figure 6b)
   - Triple Plane - For ambiguity along all axes , $M(u_1, u_2, u_3) = ([u_1; u_2; u_3], [u_1; u_2; u_3]^\intercal \in \mathbb{R}^{9\times 9}$ (Figure 6c)
3. Rotational Symmetry - This happens when the object looks identical at different points of rotation along an axis (eg. 2-fold symmetry along u_1). $M(u_1, u_2, u_3) = ([u_1], [u_2; u_3][u_2; u_3]^\intercal \in \{\mathbb{R}^{3\times 1} \times \mathbb{R}^{6\times 6}\}$
4. Axial Spherical Symmetry - Now if the rotational symmetry is along an axis with only one direction of orientation, it is termed as axial symmetry. For a glass where u_1 and $-u_1$ are not identical, $M(u_1) = u_1 u_1^\intercal$. Meanwhile in case of a cylinder, $M(u_1) = u_1$, since it appears the same at all points while rotating around the vertical axis (u_1). (Figure 6e)
5. Spherical Symmetry - This is the simplest form of symmetry and is represented by, $M = 1$. The reason being, no matter which direction a sphere is rotated about its centre, physically it looks the same. (Figure 6e)

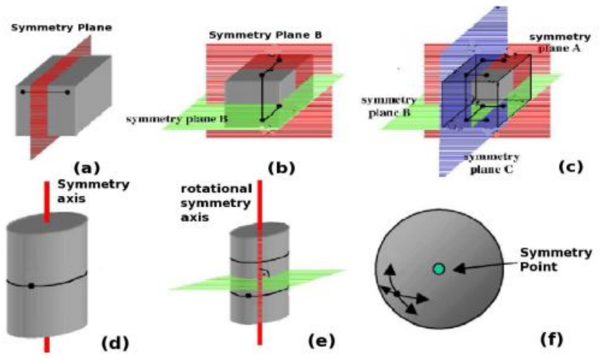

Fig. 5. 3-D Symmetries

2.2 Feature Extraction

The orientation of the objects are defined using the model described above. It is however dependent on the features. To extract the features, the gradient image of the object is considered. The authors mention of “fitting an ellipse to the edge-image of the object”. In Figure 6, this is the two circles present in the image. In turn, the image is again divided into 4 radial segments which along with the circles make up of 16 regions in the image. In each of these regions the features are obtained as the angles made by the local edges. The angles are assumed to be against one of the 2-D axes of the image in the absence of more detailed explanation.

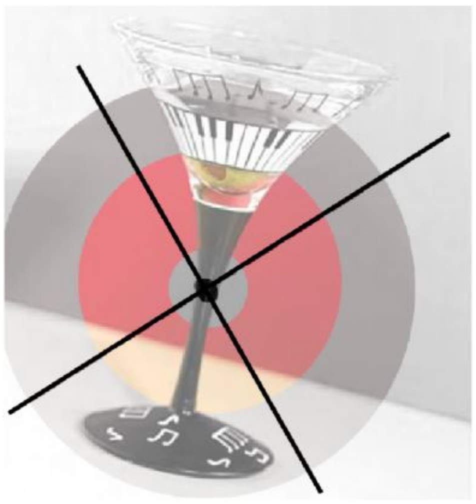

Fig. 6. Feature Extraction from Martini Glass

One important point to note is that the features and orientations need to be modelled in the same manner. In section 2.1, the models are represented as circular functions of θ. Therefore the features also need to be of the same form and dimensions. All the features from each region are then concatenated to get one feature vector. The vector can be represented as $X \in \mathbb{R}^{m \times k}$ for the orientation $Y \in \mathbb{R}^{m}$ or $X \in \mathbb{R}^{m \times m \times k}$ for the orientation $Y \in \mathbb{R}^{m \times m}$ with k number of features. This seems a bit confusing as the dimensional representation of X look similar to that of a matrix. However we assume it to be a vector as mentioned.

2.3 Inference Learning

The orientation learning model is estimated as a function of the features extracted. This is based on the multivariate Gaussian Distribution as,

$$P(Y|X;W,K) = \frac{exp(-\frac{1}{2}Tr(K(Y-XW)^\intercal(Y-XW))}{\sqrt{|(2\pi K^{-1})^n|}} \tag{2}$$

where, X = features of the image,
W = parameters of the model,
K^{-1} = shared co-variance matrix

Applying Expectation Maximization on the above model, the training data is used to maximize $\log \prod_i P(M_i|X_i, W, K)$. This in turn can be used to infer the orientation Y, depending on it being a vector or a matrix of M. The authors also warn that this model is a deficient one as the obtained output might yield false positives. In this context, the matrix Y might not satisfy the rank 1 condition.

- For a vector, $y \in \mathbb{R}^{m\times 1}$, where $yy^\intercal = 1$, the Maximum a Posteriori of y is given by,

  $$y_{MAP} = \arg\max_y Tr(Ky^\intercal XW)$$

  This has a closed formed solution as $y = \frac{XW}{||XW||_2}$
- In case of a matrix, $Y \in \mathbb{R}^{m\times m}$, it must be positive, semi-definite and have rank 1. The MAP of Y is then defined as,

  $$Y_{MAP} = \arg\min_Y -Tr(KYXW)$$

  The arg min here is assumed to be cause of cross entropy minimization though this is not mentioned explicitly. However the problem is, the matrix form of Y leads to a non-convex optimization problem. This is relaxed by dropping the rank 1 condition to convert it into a convex problem. But then the question arises how the false positives (Y matrices without rank 1) are handled. This is again not mentioned and we assume that they are included in the output. The rotation matrix can be obtained from Y by extracting the eigenvector corresponding to the highest eigenvalue. This vector is rearranged to obtain R' from which the rotation matrix is inferred by $R = R'(R'^\intercal R')^{\frac{-}{1}2}$

The precision of the model is evaluated by the error being the difference of predicted orientation angle and actual orientation angle. Since this can be non-intuitive in higher dimensions, the authors defined a new error term as Fractional Error. It is mentioned to be fraction of uniformly sampled orientations which are equal or better than the ground truth. At a high level, this can interpreted as,

$$\text{F.E.} = \frac{\text{T-C}}{\text{T}}$$

where, T = Total no of Orientation Angles or Ground Truth,
C = No of Correctly Predicted Orientation Angles

Therefore, the value of F.E. would be 0 if all angles are predicted correctly while it would be 1 if everything is incorrect.

2.4 Results

The experiments carried out by the authors are divided into two parts. Firstly orientation is estimated by training the model on 6 object classes. Secondly using the learned predictive model, orientation of a robotic hand is set to grasp an object.

In the first part, a few images are manually labeled with the 3-D orientations of the object. Due to this task being tedious and prone to errors, the remaining 9400 images are synthetically generated and labeled automatically under different conditions like changing light, camera position etc. These were used as the training images while the model was evaluated on real data with 3-D orientation labeled. The six object classes are shown in Figure 7 and the results are provided in the table (Figure 8).

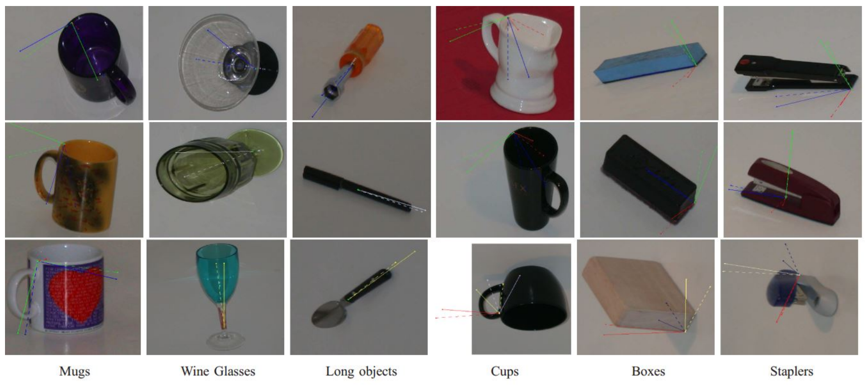

Fig. 7. Object Classes

The numbers in the table represent the actual error with the fractional error within the braces. The model is tested against other methods like the Wrapped Normal Distribution, Rotation matrix, Orientation Estimation without image features and a scaled down version of proposed method. It is evident that the proposed model outperforms all others. The error values are significantly low. Additionally the model seems to generalize well based on the example provided as it learns the orientation of a knife based on training data of a pencil. For the second part, one example is provided of a robotic hand orienting itself perpendicular to the principal axis of a pencil for gripping. In Figure 8, the last column represents the errors for this task and the authors mention that 30° accuracy is enough for efficient grasping.

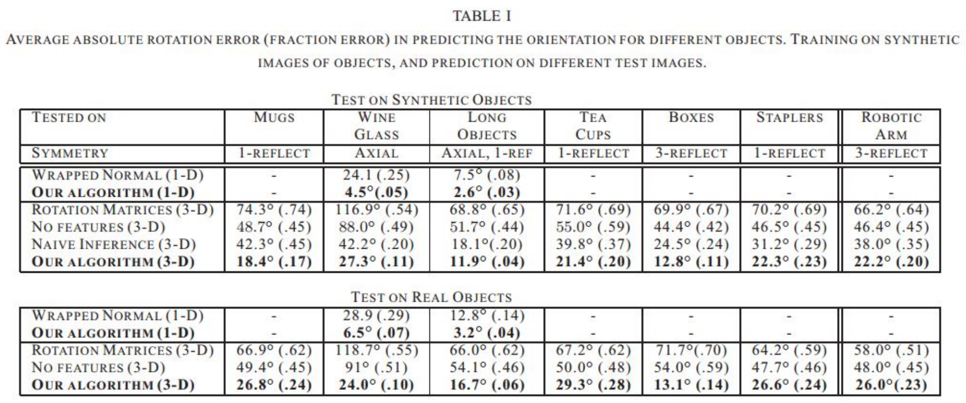

TABLE I

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)**

Fig. 8. Results

3 Discussion

The work done on estimating orientation of objects as a function of image features show that there are multiple ways to approach it. While previous to this paper, there were specific solutions for isometric or non-isometric manifold, most work failed to take into account ambiguities arising from symmetries. The paper primarily gives a solution for this and models a representation based on their own way of extracting image features.

The modelling is well represented as the authors have provided a solution which works both in Euclidean and non-Euclidean space. Moreover they are flexible for multiple types of symmetry as exhibited by the results. The comparison against other methods portray well on the proposed solution and the extra tests for robotic hand orientation looks to be working nicely. The entire section of feature learning appear a bit confusing especially w.r.t the term “angles of the local edges” (Section V-A in the paper). In the inference learning as well there is no clear explanation as to how the rotation matrix is obtained by rearranging the eigen vector i.e. in which order. One open question remains with regards to the problem of false positives (output matrix with rank not equal to 1) are handled. There is no explicit explanation provided for this either.

The number of objects and training examples look to be small taking into account the current progress made in this field. Moreover it would have been nice to have more unknown object classes as examples and the same goes for generalization.

While the threshold of accuracy is mentioned for the robotic hand orientation, there is no tests show specifically for larger objects like a wrench. Only six object classes mainly referring to indoor objects (usually found in home or office) seem naive for generalization purposes. The training data is completely based on the symmetry axes being predefined. Therefore an unsupervised learning of the axes does not seem possible with the model as proposed. This is because the model assumes which axes have ambiguity and the output would also reflect that.

As an improvement one can use the recent work by Li et al. [3] which estimates 6-D orientation using a deep neural network. Tang et al.[7] used a Dynamic Vision Sensor to determine to object orientation for better grasping and their results show the grasping accuracy to have improved to about 10°.

4 Conclusion

In conclusion, the paper introduces nice ideas in modelling orientation while reducing ambiguities from symmetry of objects. The solution is well structured and the tests look promising. However there are still some open questions regarding the feature extraction and inference learning part. Despite that, since the experimental results are encouraging, it could possibly be enhanced for more efficient representation and possibly unsupervised learning of the orientation of objects.

References

1. Dollár, P., Rabaud, V., Belongie, S.: Non-isometric manifold learning: Analysis and an algorithm. In: Proceedings of the 24th International Conference on Machine Learning. pp. 241–248. ICML '07, ACM, New York, NY, USA (2007). https://doi.org/10.1145/1273496.1273527, http://doi.acm.org/10.1145/1273496.1273527
2. Lee, C., Elgammal, A.: Modeling view and posture manifolds for tracking. In: 2007 IEEE 11th International Conference on Computer Vision. pp. 1–8 (Oct 2007). https://doi.org/10.1109/ICCV.2007.4409030
3. Li, Y., Wang, G., Ji, X., Xiang, Y., Fox, D.: Deepim: Deep iterative matching for 6d pose estimation. CoRR **abs/1804.00175** (2018), http://arxiv.org/abs/1804.00175
4. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. Science **290**(5500), 2323–2326 (2000). https://doi.org/10.1126/science.290.5500.2323, http://science.sciencemag.org/content/290/5500/2323
5. Saxena, A., Driemeyer, J., Ng, A.Y.: Learning 3-d object orientation from images. In: 2009 IEEE International Conference on Robotics and Automation. pp. 794–800 (May 2009). https://doi.org/10.1109/ROBOT.2009.5152855
6. Saxena, A., Sun, M., Ng, A.Y.: Make3d: Learning 3d scene structure from a single still image. IEEE Transactions on Pattern Analysis and Machine Intelligence **31**(5), 824–840 (May 2009). https://doi.org/10.1109/TPAMI.2008.132

7. Tang, S., Ghosh, R., Thakor, N.V., Kukreja, S.L.: Orientation estimation and grasp type detection of household objects for upper limb prostheses with dynamic vision sensor. In: 2016 IEEE Biomedical Circuits and Systems Conference (BioCAS). pp. 99–102 (Oct 2016). https://doi.org/10.1109/BioCAS.2016.7833734
8. Thomas, A., Ferrar, V., Leibe, B., Tuytelaars, T., Schiel, B., Gool, L.V.: Towards multi-view object class detection. In: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06). vol. 2, pp. 1589–1596 (June 2006)