Object Recognition with 3D Models

Bernd Heisele
bheisele@honda-ri.com
Honda Research Institute
Cambridge, USA

Gunhee Kim
gunhee@cs.cmu.edu
Carnegie Mellon University
Pittsburgh, USA

Andrew J. Meyer
ajmeyer.mit.edu
Massachusetts Institute of Technology
Cambridge, USA

Abstract

We propose techniques for designing and training of pose-invariant object recognition systems using realistic 3d computer graphics models. We look at the relation between the size of the training set and the classification accuracy for a basic recognition task and provide a method for estimating the degree of difficulty of detecting an object. We show how to sample, align, and cluster images of objects on the view sphere. We address the problem of training on large, highly redundant data and propose a novel active learning method which generates compact training sets and compact classifiers.

1 Introduction

Over the past five years, the object recognition community has taken on the challenge of developing systems that learn to recognize hundreds of object classes from few examples per class. The standard data sets used for benchmarking [11, 12] these systems contained in average less than two hundred images per class as opposed to sets of thousands of meticulously segmented object images that were used in earlier work on object detection [21, 24]. Benchmarking on such small data sets is inherently problematic, the test results are not generalizable and can be misleading; the reader is referred to [19] for a related discussion on database issues.

There have been efforts of building larger databases of manually annotated, natural images [10, 20, 25]. However, the somewhat arbitrary selection of images and the missing ground truth make it difficult to systematically analyze specific properties of object recognition systems, such as invariance to pose, scale, position, and illumination. A database for shape-based object recognition which addresses these issues is NORB [16]. Pictures of objects were taken with consideration of viewpoint and illumination. The images were synthetically altered to add more image variations, such as object rotation, background, and distractors.

Taking the idea of controlling the image generation one step further takes us to fully synthetic images rendered from realistic 3D computer graphics models. In [4, 8, 19] view-based object recognition systems have been trained and evaluated on synthetic images. The face recognition system in [14] and the object recognition system in [17] have been trained on views of 3D models and tested on real images. 3D models have also been used in a generative

approach to object recognition in which rendering parameters are optimized such that the synthetic image best matches a given photographic image [5, 26]. To avoid getting trapped in local minima, this analysis-through-synthesis approach requires a good initial estimate of the rendering parameters, making it unsuited to many object recognition/detection tasks.

In the following, we briefly discuss the pros and cons of 3D models for view-based object recognition:

+ Large numbers of training and test images can be generated fully automatically.

+ Full control over image generation parameters, including internal and external camera parameters, illumination, composition of the scene, and animation of the scene.

+ Ground truth for the location, scale, and orientation of each object. In video sequences, the frame rate, the camera motion, and the motion of objects are known.

- Lack of 3D models and 3D scenes. The situation might improve with widespread use of low-cost 3D scanners and the availability of models designed for commercial purposes (e.g., computer games, movies, Google earth 3D) and by a fast growing community of hobbyist modelers.

- Lack of realism of synthetic data. For recognition systems that operate at low pixel resolutions, the quality of synthetic image data is usually sufficient. Even at high image resolutions, free rendering software [1, 3] is capable of producing photorealistic results. However, modeling and rendering of photorealistic scenes can be time consuming processes.

- Large, accurately annotated data sets might not be necessary; humans are capable of learning to recognize new objects from a small number of images. In order to build computer vision systems with this capability, we have to better understand how humans use prior knowledge, we have to be able to implement complex visual processing steps (such as segmentation and shape from shading), and we have to understand the top-down learning mechanisms in the visual cortex.

In our first experiment, we will show that learning from 3D models requires techniques that can build compact classifiers from large, highly redundant data sets. We will suggest a method for visualizing and quantifying the degree of difficulty of detecting objects and propose a technique for aligning and clustering of images on the view sphere. Finally, we address the problem of computing compact classifiers and propose an active learning method in which the generation of synthetic training samples is controlled within an iterative training process.

2 Experimental setup

We used five textureless 3D models each with 30,000 surface triangles; they were printed on 3D printers in order to be able to test our systems on real images. The five computer graphics models and photographes of the printed models are shown in the top row of fig. 1.

The synthetic training images were rendered in Blender [1] which was interfaced through Python [2]. A camera was moved on a sphere around the model, pointing towards the model's center. The model was illuminated by ambient light and a point light source which could be positioned on a sphere around the model. The six free rendering parameters were the

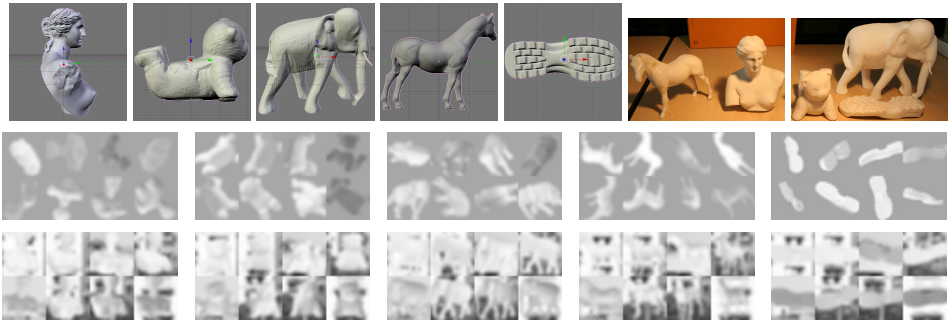

Figure 1: Top row: 3D computer graphics models and photographes of the real objects. Middle row: Synthetic images with uniform background. Bottom row: Synthetic images with natural backgrounds.

camera's location in azimuth and elevation, its rotation around its optical axis, the location of the point light source in azimuth and elevation, and the intensity ratio between ambient light and the point light. The models were rendered at a resolution of 100×100 pixels. We scaled and smoothed the rendered images resulting in 23×23 pixel grayvalue images, examples of which are shown in the middle and bottom rows of fig. 1. We decided on this low resolution because our goal was to develop a system capable of recognizing objects at low resolution, and a system sufficiently fast to be demonstrated on live video; example videos of the system applied to real videos can be found on the main author's homepage. After initial experiments with three types of features: grayvalues, normalized grayvalues, and histograms of gradients, we decided to use histograms of gradients [9, 13] for their superior performance. The histograms were computed at five fixed locations[1] within a 23×23 image resulting in a 640 dimensional feature vector which was normalized to unit length. We used an SVM [7] with a Gaussian kernel as our main classifier[2].

3 How difficult is pose-invariant object recognition?

Of course, this question cannot be answered without considering the properties of the data—such as the type of object, the pixel resolution of the object, and the image noise—and the particulars of the recognition system—such as the feature extraction algorithm and the classifier. However, by choosing a basic system composed of state-of-art components, we hope that the conclusions drawn from our experiments will be valid beyond our particular setup.

The first set of experiments dealt with the pose-invariant discrimination between two objects. We centered the objects within the training images and set the background to a fixed grayvalue. For each of the five objects we rendered 40,000 training and 20,000 test images by randomly drawing samples from the six dimensional space of rendering parameters. SVMs were trained and tested on all pairs of objects and compared to nearest neighbor classifiers. Fig. 2 shows the ROC curves for three object pairs computed on training sets with sizes

[1] The 128 dimensional histograms [13] were computed at (x/y) locations: $(9/9)$, $(15/9)$, $(12/12)$, $(9/15)$, $(15/15)$.

[2] SVM parameters were optimized in initial experiments and then kept fixed throughout: $\sigma = 2.0$, $C = 10$.

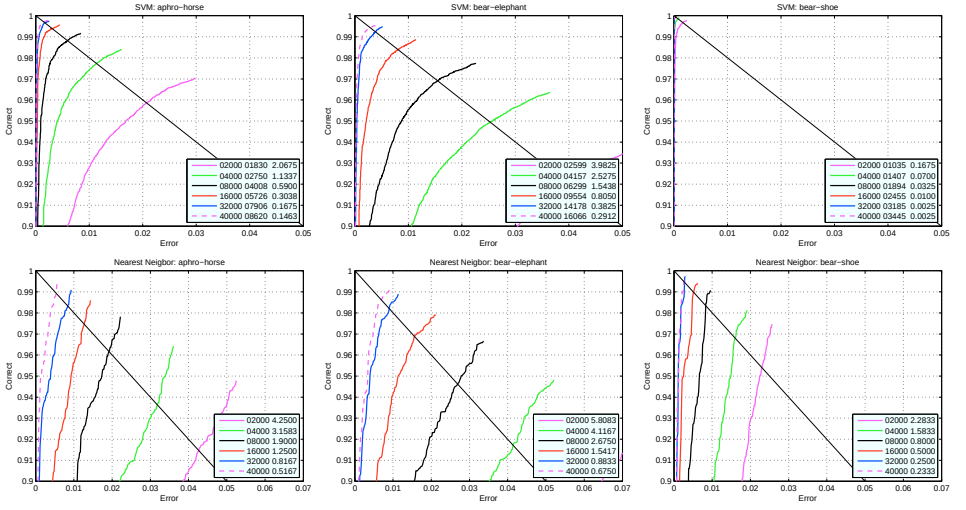

Figure 2: ROC curves for pairwise object recognition: SVM top, nearest neighbor bottom. The legends contain the number of training samples per class, the total number of support vectors, and the equal error rate in %.

between 2,000 and 40,000 samples per class. In all cases, even the ones involving the shoe sole which looked very different from the other four objects, the best results were achieved with the largest training sets. In the bear-vs.-elephant pair, for example, the equal error rate (EER) for the SVM classifier dropped from 4% at 2,000 training samples per class to 0.3% at 40,000 training samples per class. The large number of support vectors, around 16,000 for the bear-vs.-elephant SVM, and the poor recognition rates of the nearest neighbor classifiers, in particular the ones trained on the small training sets, are further indicators that the learning tasks, simple as they might have seemed initially, are non-trivial and require large training sets.

The second set of experiments dealt with pose-invariant object detection. We tried to visualize and quantify the degree of difficulty of detecting views of an object on the view sphere. In our setup, detection and recognition are *not* fundamentally different; the only difference between the two is that one of the classes in the detection task contains images of background patterns instead of images of a specific object. To locate an object of unknown size and position in a large image, our detection classifier has to be applied to a 23×23 classification window which is continuously slid across scaled versions of the input image.

We rendered views of each object on a geodesic grid with 642 vertices[3]. From these views we generated two sets of object images. In the first set, the background was set to a uniform grayvalue like we did in the previous recognition experiment. In the second set, the uniform background was replaced by natural background patterns which were cropped from a large collection of photos of various contents; some examples with natural backgrounds are shown in the bottom row of fig 1. The cropping of background patterns occurred at random image locations, random scales, and random orientations. The orientation of the crop box was randomized to avoid an overrepresentation of horizontal and vertical textures.

[3]We generated one view per vertex; the views were aligned to a reference view by a 3D alignment algorithm which will be described later in the paper.

We applied five different backgrounds to each view on the geodesic grid. The image set of the non-object/background class was composed of 20,000 background patterns of size 23×23 pixels which were cropped from photos in the same way as the background patterns for the object views.

To quantify the difficulty of detecting a given view of an object we computed the mean Euclidean distance in the feature space to its nearest neighbors[4] in the background class, i.e., we based our measure only on the most difficult examples in the background set. This 'worst-case' analysis is sensible for detection systems using the sliding window technique, where the classifier typically encounters millions of background patterns in a single image.

The experimental results for two objects are shown in fig. 3. For the set with five backgrounds per view, we averaged over the five mean distances to compute a single value per view. There are three interesting observations:

- The results differ substantially between objects and between views of the same object. Knowing which objects and which views are difficult to detect can be used for classifier design, e.g., by choosing higher pixel resolutions, more training samples, richer features, or more complex classifiers for 'difficult' objects, or 'difficult' regions on the view sphere.

- The shoe sole which was the easiest of the five objects to get recognized in the object-vs.-object task is according to our criterion the most difficult object to get detected. This shows that it is important to keep the task and the specifics of the database in mind when evaluating recognition systems. For example, a system which can distinguish between images of drinking straws and images of cars, blimps, and binoculars might not be capable of detecting a drinking straw in a picture of a dining table.

- The results for objects in front of uniform backgrounds and the results for objects in front of natural backgrounds are qualitatively very similar, indicating that the background issue can be left aside in a comparative analysis. It does not mean, however, that adding natural backgrounds becomes obsolete for classifier training.

4 Clustering of views on the view sphere

A sensible approach to pose-invariant object recognition is to break down the complex classification problem into simpler sub-problems by training separate classifiers on regions of the view sphere—an idea related to biological studies on view-tuned cells in the visual cortex, see [6]. One way to compute these regions is to apply clustering to the entirety of object images rendered across the full view sphere. Before we cluster, however, we have to consider that object views are subjected not only to rotation in depth, i.e., their location on the view sphere, but also to in-plane rotation and illumination. In the following we will focus on the problem of how to separate rotation in depth from in-plane rotation for pose-based clustering in the non-trivial case where the extracted image features are not invariant to in-plane rotation[5]. A separation, justified by the fact that in-plane rotation, unlike rotation in depth,

[4] In our experiments, the results were qualitatively similar for nearest neighbor numbers between five and twenty.

[5] The orientation histograms proved to be fairly insensitive to changes in illumination. We therefore kept the illumination fixed for the clustering experiment.

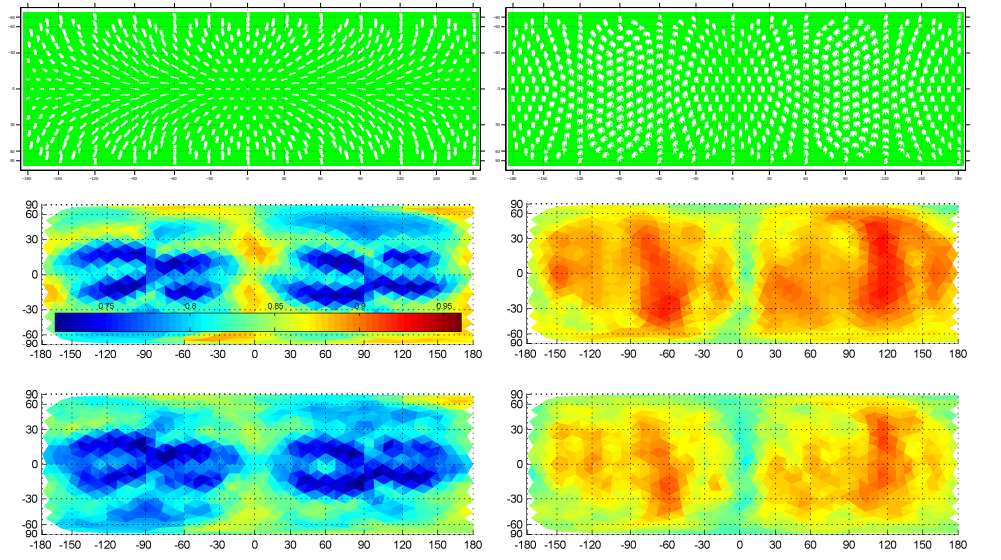

Figure 3: Object views rendered on a geodesic grid (first row). Average distance to the ten nearest background patterns in the feature space for views with uniform background (second row) and views with natural background (third row).

leaves the image contents unchanged which is also the reason why the former is considered the easier problem when it comes to pose invariance.

We studied three possibilities of removing the effects of in-plane rotation for pose-based clustering:

(1) Normalize the in-plane rotation based on statistics computed from the rendered views. The benefit of this approach is that it does not require 3D information. We computed the dominant orientation in an image from its histogram of gradients and then rotated the image accordingly, a method that was originally proposed in [18] for rotation-invariant features. We rejected this method because of its inaccuracy.

(2) Choose the in-plane rotation such that the normal vector to the (virtual) ground plane is projected onto a vertical in the image [8]. The idea is to select a 'natural' in-plane rotation, assuming that objects have a preferred orientation w.r.t. the ground plane and that photographers tend to align the camera according to the normal direction of the ground plane. The drawbacks are that the alignment function has singularities at both poles where the normal is projected onto a point and that some objects have more than one preferred orientation w.r.t. the ground plane.

(3) Choose the in-plane rotation for a pair of object views such that the pair is aligned in 3D. A view is aligned to a reference view by rotating the camera around its optical axis such that the sum of the distances between *corresponding* model surface points in the 3D camera coordinate systems of the current view and the reference view is minimized. A novel idea which avoids the pole singularities of (2) and is more accurate than (1). The rotation angle α is found by minimizing:

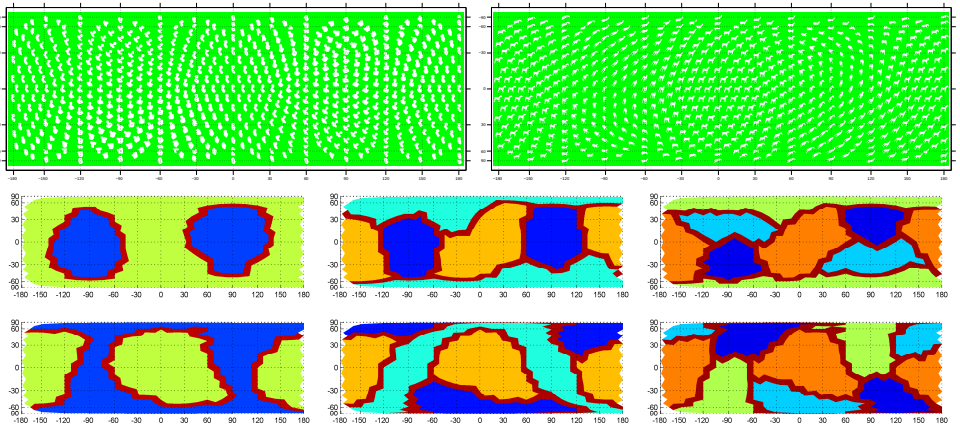

Figure 4: Results of pose-based clustering for the aphrodite (second row) and horse (third row) models.

$$\min_\alpha \sum_i \| \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix} \mathbf{x}_i - \hat{\mathbf{x}}_i \|,$$

where $\mathbf{x}_i$ and $\hat{\mathbf{x}}_i$ are the (x, y) coordinates of the model surface point i in the two camera coordinate systems. We also allowed a flip of the x-axis of the camera system in the alignment procedure which was necessary to discover mirror symmetries/similarities, see, for example, the elephant views in fig. 3 at $-60°$ and $120°$. Although the alignment method does not provide a metric in a mathematical sense—the feature distance between two aligned views depends on which view has been chosen as the reference view—the method worked well for the practical purpose of computing similarity matrices for spectral clustering.

We chose the spectral clustering algorithm by [22] in an implementation by [13], combined with the pairwise alignment procedure described above in three. The results for the aphrodite and horse models are shown in fig. 4. The clusters tend to form connected regions on the sphere although the location of the views on the sphere was not used as an input to the clustering algorithm. A naïve clustering of views based on their spherical distances, however, would lead to completely different results, as can be seen in the two-cluster result for the aphrodite model where the blue cluster consists of two clearly separated regions.

5 Active Learning

The results of our first set of experiment on pose-invariant discrimination between two objects show that the training of view-based classifiers from 3D models faces two problems:

- Large training sets which can break the learning algorithm.
- The solutions are not sparse, the classifiers will be slow at run-time.

Our 'active learning' method addresses these two problems. The term active learning in the machine learning community refers to learning from data with hidden labels. We use it to refer to a learning method that actively generates its own training data. The basic idea

is to find valuable/informative object views in the low-dimensional rendering space, six-dimensional in our experiments, and add them iteratively to the training set. Rather than bootstrapping the classifier by adding samples from a given database of images [23], we *generate* new views in each iteration.

The method consists of four steps:

(1) Train a classifier on a set of randomly selected samples. We started with 200 samples per class.

(2) Find local minima of the classifier's output in the low-dimensional rendering space.

(3) Render images at the local minima and add them to the training set.

(4) Retrain the classifier on the new training set and repeat the procedure starting from step two.

The critical part of the algorithm is step two: the search for local minima. We computed the classifier's output[6] on the same sets of 40,000 views per class that were used in our initial recognition experiment. We picked the one hundred most difficult views from each class as starting points of the Nelder-Mead simplex algorithm [15], note that we knew the values of the six rendering parameters for every view. We computed ten iterations of the Nelder-Mead simplex algorithm to find local minima of the classifier's output in the rendering space; some examples of the initially selected views and the views at the nearby local minima are shown in fig. 5. We rendered the two hundred object views at the local minima, computed their orientation histograms, and added them to the existing training set.

The results for the aphrodite-vs.-horse and bear-vs.-elephant pairs are given in fig. 5. A comparison with the results in fig. 2 clearly shows that active learning achieves the same EER with significantly smaller training sets and significantly fewer support vectors than training on a random selection of views. For example, the actively trained classifier after twenty iterations for the aphrodite-vs.-horse pair was trained on 2,100 samples per class, had a total of 3,715 support vectors, and an EER of 0.114%, compared to 40,000 training samples per class, 8,620 support vectors, and an EER of 0.146% in the initial experiment. For the bear-vs.-elephant pair, the actively trained classifier after thirty-three iterations was trained on 3,500 samples per class, had a total of 6,654 support vectors, and an EER of 0.163%, compared to 40,000 training samples per class, 16,066 support vectors, and an EER of 0.291% in the initial experiment.

6 Conclusion

We proposed several new ideas of how to use 3D models for view-based object recognition. Our first experiment showed that if high accuracy and pose invariance are to be achieved, even the seemingly simple task of distinguishing between two objects requires training sets that are far larger than any of the commonly used object recognition databases of natural images. We introduced a technique for visualizing and quantifying the degree of difficulty of detecting objects based on nearest-neighbor distances to background patterns and proposed an alignment algorithm for pose-based clustering on the view sphere. Finally, we suggested

[6] We used the real-valued output of the SVM. For samples of the negative class (label -1) the output was multiplied by -1.

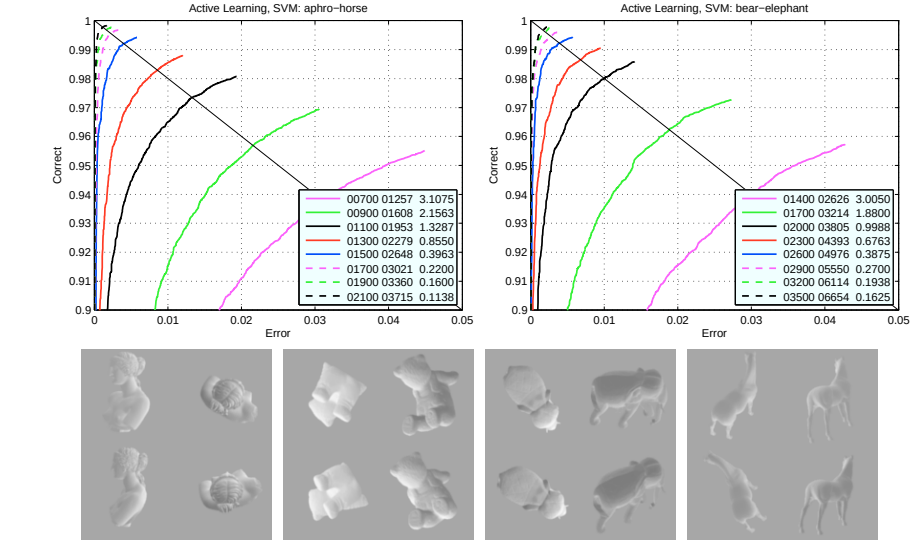

Figure 5: Top: ROC curves for pairwise object recognition with active learning. The legends contain the number of training samples per class, the total number of support vectors, and the equal error rate in %. Bottom: Example pairs of initial views and views at nearby local minima in the six-dimensional rendering space after ten iterations of the Nelder-Mead simplex algorithm. The pairs are vertically arranged with the initial views on top.

a novel learning algorithm which iteratively generates a sparse training set by searching for local minima of a classifier's output in a low-dimensional space of rendering parameters.

We believe that the use of realistic 3D models and computer graphics for view-based object recognition will lead to a reevaluation of some of the basic research questions in this area: the problems of learning from small training sets and learning from data with inaccurate or missing ground truth will lose importance while the problem of learning from large, accurately labeled but highly redundant data will become more important. Compared to learning from large data sets of natural images, learning from 3D models is substantially different since it offers the possibility of obtaining ground truth and the possibility of generating training data by controlling a synthetic image generation process through a small number of rendering parameters. None of the techniques we introduced and none of the experiments we conducted would have been feasible with natural images.

References

[1] *Blender Version 2.48a*. URL http://wiki.blender.org/index.php/Doc:Manual.

[2] *Python v2.6.1*. URL http://docs.python.org/.

[3] *YafaRay Version 0.1.0*. URL http://www.yafaray.org/documentation.

[4] V. Blanz, B. Schölkopf, H. Bulthoff, C. Burges, V. Vapnik, and T. Vetter. Comparison

of view-based object recognition algorithms using realistic 3d models. In *International Conference on Artificial Neural Networks (ICANN)*, pages 251–256, 1996.

[5] V. Blanz, S. Romdhani, and T. Vetter. Face identification across different poses and illuminations with a 3d morphable model. In *Proc. Conference on Automatic Face and Gesture Recognition (FG)*, pages 202–207, 2002.

[6] E. Bricolo, T. Poggio, and N. Logothetis. 3d object recognition: A model of view-tuned neurons. In *Neural Information Processing Systems (NIPS)*, pages 41–47, 1996.

[7] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[8] C. M. Cyr and B. B. Kimia. 3d object recognition using shape similarity-based aspect graph. In *International Conference on Computer Vision (ICCV)*, pages I: 254–261, 2001.

[9] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 886–893, 2005.

[10] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2008 (VOC2008) Results. http://www.pascal-network.org/challenges/VOC/voc2008/workshop/index.html.

[11] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. 2004.

[12] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology, 2007. URL http://authors.library.caltech.edu/7694.

[13] M. Hein and U. von Luxburg. *GraphDemo: a Matlab GUI to explore similarity graphs and their use in machine learning*. URL http://www.ml.uni-saarland.de/GraphDemo/GraphDemo.html.

[14] B. Heisele and V. Blanz. *Face Processing, Advanced Modeling and Methods*, chapter 14, pages 439–462. Elsevier, 2006.

[15] J. C. Lagarias, J. A. Reeds, M. H. Wright, and P. E. Wright. Convergence properties of the nelder-mead simplex method in low dimensions. *SIAM Journal of Optimization*, 9 (1):112–147, 1998.

[16] Y. LeCun, F. J. Huang, and L. Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 97–104, 2004.

[17] J. Liebelt, C. Schmid, and K. Schertler. Viewpoint-independent object class detection using 3d feature maps. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2008.

[18] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[19] N. Pinto, D. Cox, and J. DiCarlo. Why is real-world visual object recognition hard? *PLoS Computational Biology*, 4(1):e27, 01 2008.

[20] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. Labelme: A database and web-based tool for image annotation. Technical report, Massachusetts Institute of Technology, 2005.

[21] H. Schneiderman and T. Kanade. A statistical method for 3D object detection applied to faces and cars. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 746–751, 2000.

[22] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:888–905, 2000.

[23] K. K. Sung and T. Poggio. Example-based learning for view-based human face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(1):39–51, 1998.

[24] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 511–518, 2001.

[25] L. von Ahn and L. Dabbish. Labeling images with a computer game. In *Proc. SIGCHI Conference on Human Factors in Computing Systems*, pages 319–326, 2004.

[26] P. Yan, S. M. Khan, and M. Shah. 3d model based object class detection in an arbitrary view. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, pages 1–6, 2007.