Learning a Probabilistic Model Mixing 3D and 2D Primitives for View Invariant Object Recognition

Wenze Hu[⋆,‡], and Song-Chun Zhu[⋆,‡]
[⋆]Department of Statistics, University of California, Los Angeles (UCLA), USA
[‡]Lotus Hill Research Institute (LHI), Ezhou, China
{wzhu, sczhu}@stat.ucla.edu

Abstract

This paper presents a method learning mixed templates for view invariant object recognition. The template is composed of 3D and 2D primitives which are stick-like elements defined in 3D and 2D spaces respectively. The primitives are allowed to perturb within a local range to account for instance variations of an object category. When projected onto images, the appearance of these primitives are represented by Gabor filters. Both 3D and 2D primitives have parameters describing their visible range in a viewing hemisphere. Our algorithm sequentially selects primitives and builds a probabilistic model using the selected primitives. The order of this sequential selection is decided by the information gains of primitives, which can be estimated together with the visible range parameter efficiently. In experiments, we evaluate performance of the learned 3D templates on car recognition and pose estimation. We also show that the algorithm can learn intuitive mixed templates on various object categories, which suggests that our method could be used as a numerical method to justify the debate over viewer-centered and object-centered representations.

1. Introduction and Related Works

In the literature, representations for view invariant object recognition can be categorized to two alternatives: viewer-centered representation [8, 15] and object-centered representation [1, 14]. The two methods are usually studied separately. There were debates [2, 22, 5] arguing which one is better, but researchers did not draw final conclusion.

This paper presents a model mixing both representations via 3D and 2D primitives, and a method learning mixed templates for view invariant object recognition. By automatically selecting primitives, this method suggests a numerical method to justify the best representations for different object categories.

Images of desktop globe in Fig.1 serve as an example to illustrate our model. Our model chooses 2D primitives to explain the images of the globe part, because it is 2D circular shape across all views. For the handle and base of the desktop globe, our model selects 3D primitives to form their 3D shapes. This is because their image patterns change across views, but are all projections of same 3D shapes.

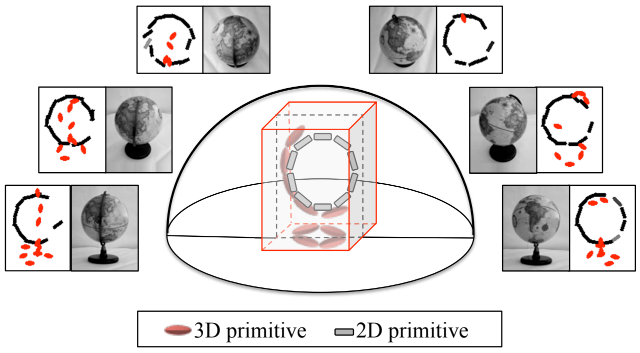


Figure 1. Illustration of learned templates of a desktop globe (Better viewed in color). Templates have 3D primitives and 2D primitives, which are denoted in red and gray respectively. Our algorithm can automatically select 3D primitives to describe globe's base and handle, whose appearances vary across views, and 2D primitives to describe the occlusion boundary of the globe, which is a circular pattern across all views.

Inspired by active basis model [25] and recent work on learning both object texture and structure[18], we use template containing both 3D and 2D primitives to represent images of an object category from different views. The 3D and 2D primitives are stick-like elements defined in 3D and 2D spaces. To model object image variation, primitives are allowed to perturb within a small location and orientation range. We use a visible range parameter to model 3D primitive occlusion and 2D primitives only seen on part of the view space. On images, the appearance of these primitives are represented by Gabor filters. Position and orientation of Gabor filter for a 3D primitive change across different views.

With the mixed representation, we build up our genera-

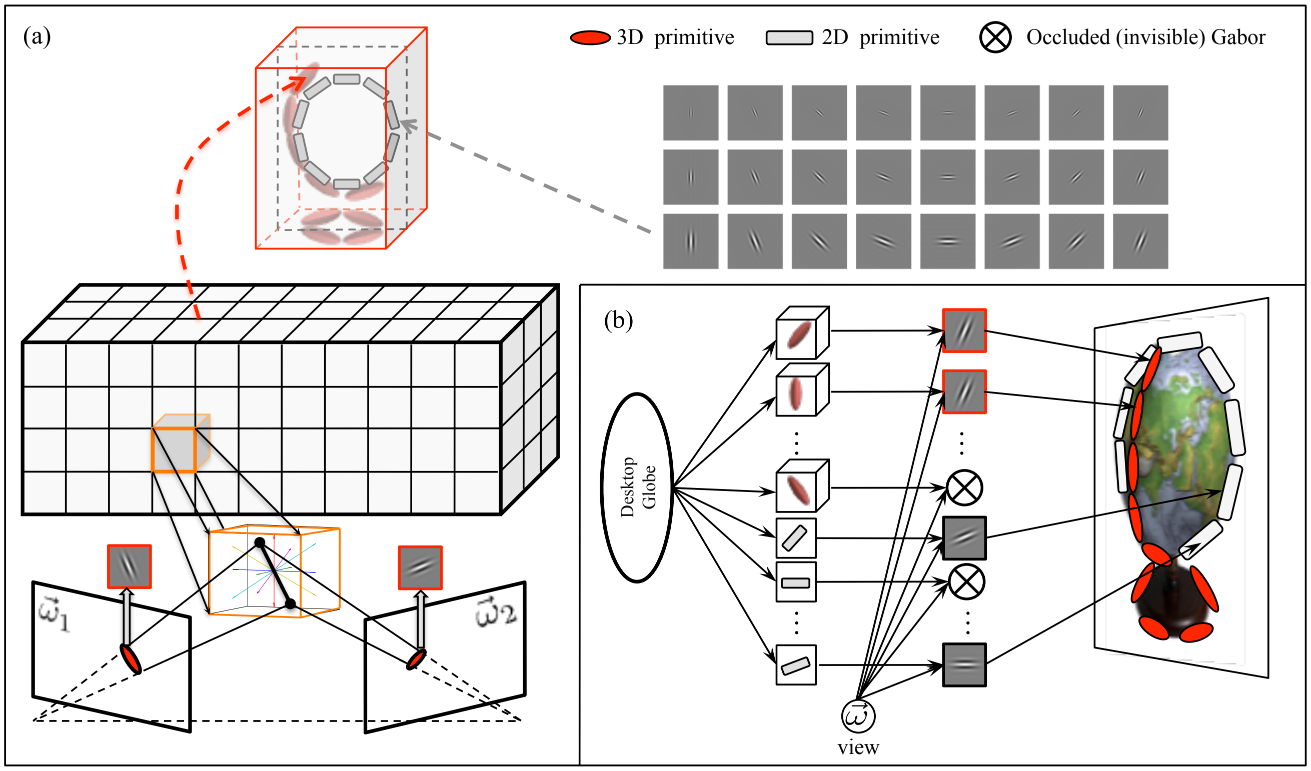


Figure 2. Overview of our model (Better viewed in color). (a) Illustration of 3D and 2D primitives and how they are used to compose mixed representation. The 3D primitives are shown on the lower-left part. A 3D primitive can be viewed as a stick with selected orientation and rotation in 3D space, its appearance a Gabor filters placed at projected positions and orientations. The 2D primitives are stick-like elements located at selected 2D positions and orientations. (b) When generating object images at a particular view $\vec{\omega}$, we project the learned 3D primitives and transform the 2D primitives.

tive probabilistic model from object category images of different views. In our model, when primitive is visible, we model the filter response of primitive following maximum entropy principle [7, 16]. When it is not, we assume the response follows the distribution of Gabor filtering response on natural images.

Our model can be learned using a pursuit algorithm, which sequentially select primitives using information gain as pursuit index. The key issue in evaluating the information gain of a primitive is to estimate its visible range parameter. Using the concavity of the information gain term detailed in Section 3.1, we are able to avoid enumerating all the possible visible ranges and apply an efficient algorithm to simultaneously estimate the optimal visible range and information gain for each primitive. Selected primitives form intuitive templates as is shown in Fig.1, and can be used in tasks such as object recognition and pose inference. Also, learned templates from various object categories suggest that our method could be used as a numerical way to justify the debate over viewer-centered representation and object-centered representation.

In recent literature, simultaneously learning 3D object shape and appearance is successfully implemented in [9, 4] for multi-view images, but only for a single object. For images from different instances of a category, recent work either take the 3D shape as granted and directly learn the appearance model [6, 13] , or learn shape and appearance one after another [26]. Our model is different from those works since we can learn 3D shape as well as appearance simultaneously.

Recent work on viewer-centered representation are usually based on existing single view object recognition methods. They proposed to link 2D features [23] or find shared features [24] across views to build efficient model. Researchers also proposed to build up image patches composed of multiple feature points as an intermediate level of object category representation, add links and transformations between these patches [10, 17, 21], or add hierarchical models on them [20] to build better model. These models successfully extended current work on single view object recognition, but did not fully exploit 3D information of objects.

The main contributions of this paper are: i. We propose 3D primitive based object representation together with algorithms to learn templates of view variant images from an object category. ii. By using information gain as a criteria to pursue both 3D and 2D primitives, our algorithm can automatically select primitives from both representations, which provides a numerical method to justify the debate over object-centered and viewer-centered representation.

2. Representation and Model

As is shown in Fig.2, image is represented by primitives chosen from a dictionary of 3D and 2D primitives. We use Gabor filters as appearance of these primitives. In this paper, Gabor filters at different position and orientation are treated as different filters. Currently, we only use one scale of Gabor filters in our implementation.

2.1. A dictionary of 3D and 2D primitives

(a) 3D primitives. To represent 3D shape of objects, we propose stick-like 3D elements as primitives. We define its geometric parameter in 3D space by its position (X, Y, Z) and orientation $\vec{\Theta}$. Currently, we fix the length of 3D primitives, which corresponds to the fixed size of Gabor filters. With these parameters, the set of 3D primitives can be expressed as $\{B_{X,Y,Z,\vec{\Theta}},\ \forall X, Y, Z, \vec{\Theta}\}$. For a 3D primitive, the position and orientation of its Gabor filter is decided by its 3D to 2D projection. Hence, as is shown in lower-left part of Fig.2, the appearance of a 3D primitive will be different for different views.

In this paper, a view $\vec{\omega}$ is expressed as a vector with its elements pan, tilt, roll, scale and object image center offset relative to the center of the whole image. By the definition of view, the view space in this paper is the space spanned by the elements in view vector.

We use orthogonal projection as our projection model. When projecting primitives to object images, some of them may be occluded. So we use a parameter Ω to denote the views in which the corresponding primitive is visible. Ω is defined as a collection of views in the view space. For example, $\Omega = \{\vec{\omega}_k, k = 1, 2, \cdots, K\}$, where K is the number of views.

(b) 2D primitives. The 2D primitives b used in this paper are stick-like elements defined in 2D space. This set of primitives can be expressed as $\{b_{x,y,\theta},\ \forall x, y, \theta\}$, where x, y are indexes of 2D positions and θ is the index for orientation. The 2D primitives are also associated with visible range parameter Ω, and is used to describe the views in which a 2D primitive appears. This Ω has the same form as the one for 3D primitives.

For both 3D and 2D primitives, we define primitive response on image as the response of corresponding Gabors. We allow the Gabors to locally move along its direction and rotate among its neighboring directions, and take the maximum of filtering response as the response of current Gabor. Gabor filter response is defined as the squared norm of inner product between the Gabor filter and image.

In the following, we will first introduce definition of image generating model, image probability model and learning algorithm for 3D primitives. We then will show that this algorithm can also be applied to our 2D primitives, and hence becomes a template learning algorithm mixing object-centered representation and viewer-centered representation.

2.2. Probabilistic Image Model

Consider the case that our object representation is composed of only N 3D primitives $\{B_i, i = 1, \cdots, N\}$ with corresponding visible range $\{\Omega_i, i = 1, \cdots, N\}$, where we use i instead of $X, Y, Z, \vec{\Theta}$ to index the selected subset of primitives. For a given view $\vec{\omega}$, the object image $\mathbf{I}$ can be expressed as:

$$\mathbf{I}|\vec{\omega} = \sum_{i=1}^{N} \alpha_i B_i \cdot \mathbf{1}(\vec{\omega} \in \Omega_i) + U, \tag{1}$$

where α_i is the coefficient associated with each primitive, U is unexplained part of image, and $\mathbf{1}(\cdot)$ is the indicator function.

Following the derivation of Active Basis model [25], the target image distribution can be written as:

$$p(\mathbf{I}|\vec{\omega}) = q(\mathbf{I}) \prod_{i=1}^{N} \frac{p(r_i|\vec{\omega})}{q(r_i)} \tag{2}$$

where $p(r_i|\vec{\omega})$ refers to the response distribution of i-th 3D primitive, $q(\mathbf{I})$ is the distribution of reference images. $q(r_i)$ is primitive response distribution on reference images, which should be independent of $\vec{\omega}$. For readers unfamiliar with Active Basis model, a random collection of natural images is used in the model and called reference images. In practice, $q(r_i)$ should be the same for all primitives. To save computation, we can pool a general $q(r)$ over these reference images and store it as a histogram.

Whether a 3D primitive is visible or not significantly affects the distribution of its observed primitive response. Hence, when modeling $p(r_i|\vec{\omega})$, we model the case of primitive is visible and occluded separately.

When the primitive is visible, according to maximum entropy principle [7, 16], the response can be modeled as $p(r_i; \lambda_i) = {Z_i}^{-1} \exp(\lambda_i r_i) q(r_i)$. When it is not, the observed response is assumed to follow reference distribution, thus $p(r_i) = q(r_i)$. Taking the visible range parameter into account, $p(r_i)$ can be expressed as:

$$p(r_i; \lambda_i, \Omega_i|\vec{\omega}) = \begin{cases} \frac{1}{Z_i} \exp(\lambda_i r_i) q(r_i) & \vec{\omega} \in \Omega_i \\ q(r_i) & \vec{\omega} \notin \Omega_i \end{cases} \tag{3}$$

To ensure that Eqn. (3) fits target distribution, λ_i should be estimated subject to constraints that given K visible re-

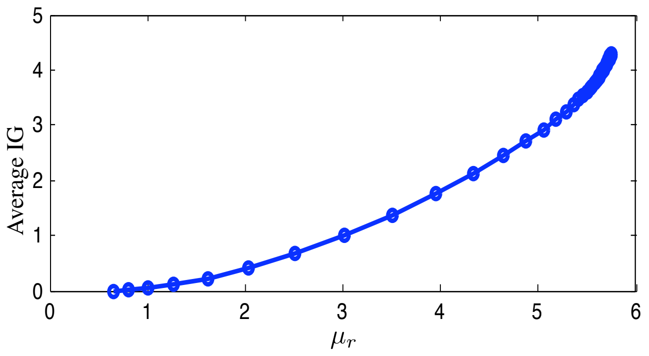


Figure 3. The relationship of average information gain against μ_r. This curve shows that the average information gain is an increasing function of μ_r. The reason that why μ_r is in range $[0, 6]$ is explained in Section 5.1.

sponses $\{r_{ik}\}_{k=1}^{K}$,

$$\frac{1}{K}\sum_{k=1}^{K} r_{ik} \approx \int r_i \frac{1}{Z_i}\exp(\lambda_i r_i) q(r_i) \mathrm{d}r_i \tag{4}$$

3. Learning Algorithm

At learning stage, given a collection of M view labeled images $\{(I_m, \vec{\omega}_m), m = 1 \cdots M\}$, our goal is to find a set of primitives together with parameters $\{\lambda_i, \Omega_i\}$ that maximize the likelihood of Eqn.(2).

The learning algorithm first computes the responses of all Gabor filters on all training images. Specifically, we compute responses of these filters centered at each pixel of an image, and at 15 equally spaced orientations. Following [25], we call these filtered responses together as SUM1 maps. For each Gabor filter, we compute its maximum responses in neighboring locations and orientations, and save these maximized responses in so called MAX1 maps.

As the same with learning tasks in [25, 16], this primitive selection problem can be solved under the matching pursuit framework, which incrementally select primitives from a large set of candidates to approximate the target distribution. We will first explain the key steps of this learning algorithm and then list all steps later.

3.1. Evaluate the Information Gain of a Primitive

Given a 3D primitive and M training images, the information gain by updating i-th primitive's distribution from reference distribution to target distribution is

$$\sum_{m=1}^{M} \log \frac{p(r_{im}|\omega_m)}{q(r_{im})} \tag{5}$$

$$= \sum_{\vec{\omega}_m \in \Omega_i} (\lambda_i r_{im} - \log Z_i) + \sum_{\vec{\omega}_m \notin \Omega_i} 0 \tag{6}$$

$$= \sum_{\vec{\omega}_m \in \Omega_i} (\lambda_i r_{im} - \log Z_i) \tag{7}$$

If Ω_i is given, parameter λ_i should be fitted such that

$$\hat{\mu}_r \approx \frac{1}{M'} \sum_{m'=1}^{M'} r_{im'} = E_p(r_i; \lambda_i) \tag{8}$$

Where $\hat{\mu}_r$ is the mean response of samples in view range Ω_i, and M' is the number of of these samples. To reduce computation, one can compute a histogram of $E_p(r; \lambda)$ indexed by λ before learning starts. In learning stage, the best λ can be get by searching the histogram and interpolate between the nearest two bins.

In the above derivation, we assume Ω_i is given, but our objective is to maximize the information gain on both λ_i and Ω_i. Remember that we set Ω_i as a collection of views. So, given M training images, we can enumerate all the 2^M different Ω_i, find the corresponding maximum information gain for each one, and hence find the global optimal.

Though the method above gives correct result, the computation is way too expensive. By re-arranging the elements in Eqn.(7), maximizing Eqn.(7) is equivalent to:

$$\max \ |\Omega_i|(\lambda_i \cdot \hat{\mu}_r - \log Z_i) \tag{9}$$

where $|\Omega_i|$ is the number of training views in Ω_i. Eqn.(9) indicates that according to the number of views in Ω_i, we can put the 2^M possible Ω_is into M groups. Benefits of grouping them are led by the following observations: **1.)** For Ω_is in each group, $|\Omega_i|$ is fixed, maximizing Eqn (9) is reduced to maximize the average information gain $(\lambda_i \cdot \hat{\mu}_r - \log Z_i)$; **2.)** For log linear model, it has been proved that this term is an increasing function of $\hat{\mu}_r$. The second observation is proved in [25], and here we just show the corresponding curve in Fig.3 as a validation.

According to these two observations, in a group of Ω_is having t views, optimal Ω_i must be the one that leads to largest $\hat{\mu}_r$, which should be the mean of the t largest of totally M responses.

In implementation, we can avoid enumerating 2^M combinations and instead sort the primitive responses, induce the optimal Ω_i in each group and compare the best Ω_is to find global optimal. Int this way, we can reduce the 2^M computations to at most M. The corresponding algorithm is shown in Algorithm (1). Note that the step 6 is done in the similar way of computing λ. The break at step 11 is because our objective function is a concave function, which could be easily proved following chapter 3 of [3].

Algorithm 1: Evaluate information gain of a primitive

```
Input: ${r_{im}, \vec{\omega}_m}_{m=1}^M$
Output: Estimated parameter $\widehat{\lambda}_i, \log Z_i, \widehat{\Omega}_i$,
        Information gain $IG_i$
1  $IG^{\max} \leftarrow 0, t^{\max} \leftarrow 0$
2  $\{r'_{im}, \vec{\omega}'_m\}_{m=1}^M \leftarrow$ descending sort on $r_{im}$
3  for $t \leftarrow 1$ to $M$ do
4      $\mu^t \leftarrow$ mean$(r'_{i1}, \cdots, r'_{it})$
5      $\lambda^t \leftarrow$ interpolate table $E_p(r;\lambda), s.t.$
       $\mu^t \approx E_p(r;\lambda^t)$
6      $\log Z^t \leftarrow$ search table $\log Z(\lambda)$ to get $\log Z(\lambda^t)$
7      $IG^t \leftarrow t \cdot (\lambda^t \mu^t - \log Z^t)$
8      if $IG^{\max} < IG^t$ then
9          $\widehat{\lambda}_i \leftarrow \lambda^t, IG^{\max} \leftarrow IG^t, t^{\max} \leftarrow t$
10     else
11         Break;
12     end
13 end
14 $\log Z_i \leftarrow \log Z(\widehat{\lambda}_i), \widehat{\Omega}_i \leftarrow \{\vec{\omega}'_k\}_{k=1}^{t^{\max}}, IG_i \leftarrow IG^{\max}$
15 return $\hat{\lambda}_i, \log Z_i, \widehat{\Omega}_i, IG_i$
```

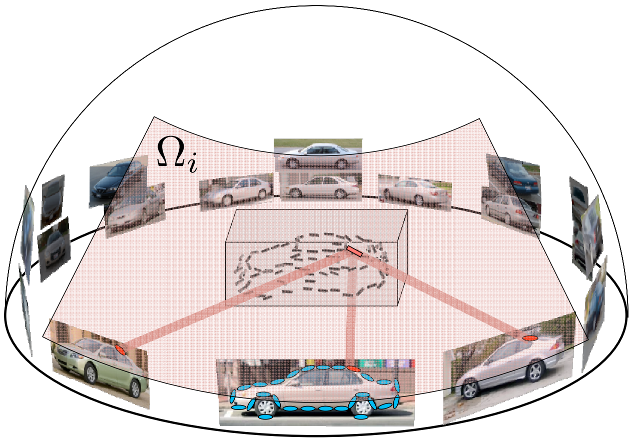


Figure 4. Illustration of how 3D primitives are learned from and projected to images in different views. The learning step can be interpreted as trying all possible locations and orientations of 3D primitives and keep the independent ones with large projected responses. In testing stage, the learned 3D model is projected to each possible view and check if the projected template matches testing image.

3.2. Primitive Pursuit

Using the estimated information gain, we can start to build up our statistical model by pursuit algorithm.

An Intuitive explanation of how primitive is selected is shown in Fig.4. Before pursuit, we enumerate lots of possible primitives in 3D space. As is shown in Fig.2, we propose 3D primitives centering at evenly sampled positions of a cuboid. For each sampled position, we place primitives at evenly spaced orientations using code from [12].

We then compute each 3D primitive's responses by referencing the projected Gabor responses on MAX1 maps. After that, we can use Algorithm (1) to compute each primitive's information gain, and then select the primitive with maximum information gain. At the same time, parameters associated with primitive is also stored. We further locally inhibit the selected Gabor responses, which serves to avoid selecting similar primitives. Details of this inhibition step can be found in [25].

After the first primitive is selected, we repeat the process starting from updating the information gain of each primitive so as to sequentially pursue a collection of primitives from the enumerated primitive pool. The procecdure of this algorithm is summarized in Algorithm (2).

Algorithm 2: Primitive pursuit

```
Input: $M$ training images $\{I_m, \vec{\omega}_m\}$
Output: N selected primitives with parameter
        $[(X_i, Y_i, Z_i, \vec{\Theta}_i), \lambda_i, \log Z_i, \Omega_i]$
1  Compute SUM1 maps and MAX1 maps.
2  Enumerate all the possible primitives.
3  Project each primitive on to the MAX1 maps.
4  for $i \leftarrow 1$ to $N$ do
5      Retrive primitives' responses from MAX1 maps.
6      for each primitive do
7          compute $IG_i$, using algorithm 1.
8      end
9      Find the maximum $IG$, denote it as $IG_{\max}$
10     Retrieve and store parameters associated with
       $IG_{\max}$
11     $(X_i, Y_i, Z_i, \vec{\Theta}_i) \leftarrow (X_{\max}, Y_{\max}, Z_{\max}, \vec{\Theta}_{\max})$
12     Local inhibition.
13 end
```

Note that the during primitive pursuit, information gain of a primitive is non-increasing, hence we only need to compute IG for all primitives in the first iteration, and then delete primitives whose IG are lower than a threshold, since they will never been selected.

If we model the 2D primitive response in the same way described in Eqn. (3), the algorithm proposed above can be used to pursuit both 3D and 2D primitives. In fact, assuming observed primitive response distribution on invisible views to follow that on natural images is implicitly used in many models such as [10] in reference and has at least explicitly used in [19].

To implement the algorithm in learning mixed representation, the only step we need to further specify is how to enumerate 2D primitives. Because object image scale and offset changes across training images, we assume there is

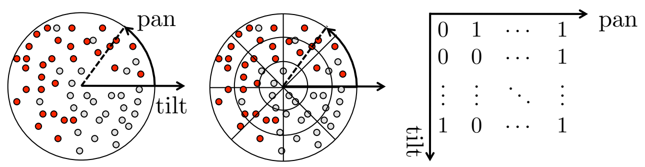


Figure 5. Convert Ω from a set to a 2 dimensional lookup table (Better viewed in color). **Left**: red points represent the views that are in Ω, gray ones are the rest. **Middle**: we segment the view sphere in to several view bins. **Right**: each bin corresponds to an entry of a 2D lookup table, where its value is set by the majority of the training views in that bin.

an virtual image plane, and propose 2D primitives at each position and orientation in this plane. To associate 2D primitives with image responses, the projection step used in 3D primitive is degenerated to scaling and translating the primitives.

4. Inference Algorithm

In our formulation, Ω is defined as a collection of single views. To ensure that our model can also detect views not appeared in training set, we further convert this Ω into a 0-1 lookup table. The conversion steps are shown in Fig.5. Considering the fact that in orthogonal projection, image center offset, scale, and roll angle do not affect primitives' occlusion, we only need to set up a 2 dimensional lookup table along pan and tilt angle. The entry value is set to 1 if majority ($> 50\%$) of training images in that bin is included in Ω.

We formulate the inference problem as a hypothesis testing problem. The testing score is defined as

$$\text{score}(\mathbf{I}) = \log \frac{p(\mathbf{I}|\vec{\omega})}{q(\mathbf{I})} = \sum_{i=1}^{N} (\lambda_i r_i - \log Z_i) \cdot \mathbf{1}(\vec{\omega} \in \Omega_i) \tag{10}$$

Where N is the number of selected primitives. Since Ω has already been converted to a lookup table, the value of function $\mathbf{1}(\vec{\omega} \in \Omega_i)$ can be easily computed.

To find the object instance from a testing image, we compute the testing score for lots of view points. For recognition, we only perform coarse search at step width $10°$ in pan, $5°$ in tilt and $2°$ in roll. For pose inference, we perform fine search at step width $2°$ for both pan tilt and roll.

5. Experiments

5.1. Dataset and Image Pre-processing

To show the effect of our algorithm on uncontrolled images with relatively complex background, a multi-view car dataset [17] is used. Since view labels are not provided, we labeled them buy imposing 3D CAD models onto images. To show our algorithm can learn intuitive templates for various categories, we use ETH80 dataset [11], which contains 8 categories with 10 object instances for each category. We further use soda cans as an object category, and the desktop globe images used in Fig.1. The last two sets of images are taken from LHI dataset [27].

Pre-processing steps used in experiments are: 1.) We normalize the responses on a SUM1 map by dividing them with response variance on that image. 2)We use local normalization with window size 32 pixels. 3) We further regulate responses using sigmoid function, that is:

$$r' = \text{sigmoid}(r) = \xi[2/(1 + e^{-2r/\xi})] \tag{11}$$

Where ξ is set to 6. Theoretical underpinnings of these pre-processing transformations can be found in [25]. In experiments on dataset [17], the location and orientation change for computing MAX1 maps is ± 4 pixels and ± 1 orientations. In experiment on ETH80 dataset, these are ± 3 pixels and ± 1 orientations.

5.2. Learning 3D Templates for Car Recognition and Viewpoint Inference

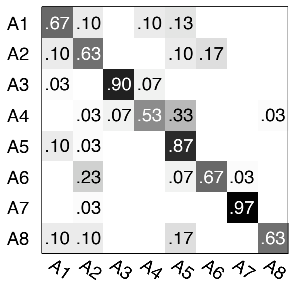
First, we use our learning algorithm to learn 3D primitives on the multi-view car dataset [17]. We use the first 5 instance as training images, and the rest 5 as testing.

We first take the experiment of object pose inference. In this paper, it is defined as finding the most probable view category given that the object image is in a bounding box. For this part, we took the images in the given bonding box as input, and search around views $\vec{\omega}$ to find the one with maximum score. The estimated view is then converted to view category as output to compute the confusion matrix. The result is shown in Fig.6 with both confusion matrix and some of template matching results. [1]

Since our algorithm's output is view angles instead of view category. It should be more informative in pose related inference tasks such as license plate searching. Also, by projecting the templates, results also approximately shows the boundaries of objects, which can be used as a good initial state for tasks like image segmentation.

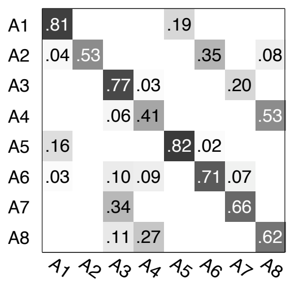
We also show object detection results on this dataset. The detection task is done by finding the maximum score defined in Eqn. (10) for each given bounding box, and we use traditional sliding window and non-maximum suppression methods to propose detection result. For each given bounding box, we search on different views, and select the one with maximum score as the score of current bounding box. Following the protocol of VOC detection task, we evaluate our algorithm's performance in terms of precision

[1]It should be clarified that this experiment is not completely the same as in [20], since 1)We test not only on correctly recognized images, but all testing images. 2) We are also testing scale 3 of the dataset.



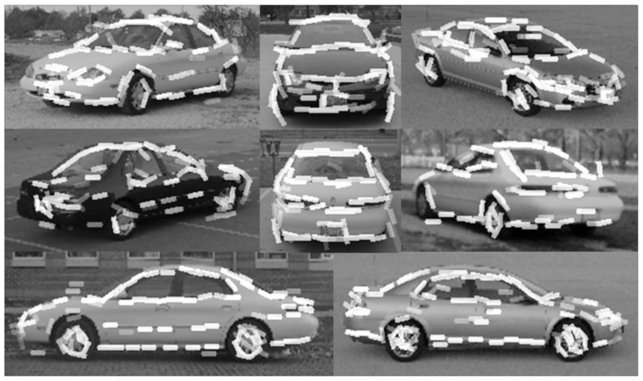
	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8
A1	.67	.10		.10	.13			
A2	.10	.63			.10	.17		
A3	.03		.90	.07				
A4		.03	.07	.53	.33			.03
A5	.10	.03			.87			
A6		.23			.07	.67	.03	
A7		.03					.97	
A8	.10	.10			.17			.63

(a)



	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_8
A1	.81				.19			
A2	.04	.53				.35		.08
A3			.77	.03			.20	
A4			.06	.41				.53
A5	.16				.82	.02		
A6	.03		.10	.09		.71	.07	
A7			.34				.66	
A8			.11	.27				.62

(b)



(c)

Figure 6. Results of pose estimation task. (a) Confusion matrix of our method. (b) Confusion matrix from [20], which we use as a reference. (c) projected templates on testing images from different views.

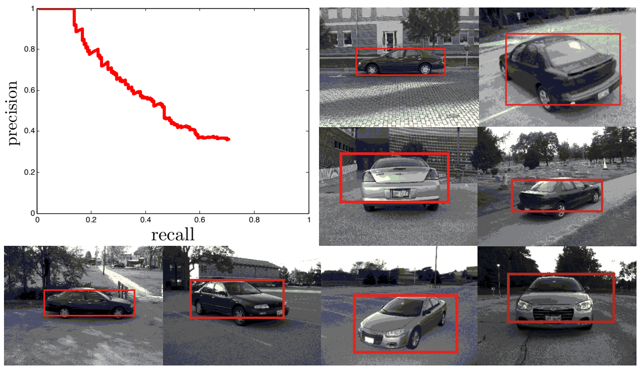


Figure 7. Performance of detection task in terms of PR Curve together with some detection results.

recall curve. The PR curve together with some detection results are shown in Fig7.

5.3. Learning Mixed Templates

To demonstrate the effect of our mixed template learning algorithm, we show the learned templates for 10 categories in Fig.8. The categories are sorted by the proportion of 3D primitives in the mixed templates, and are shown in descending order.

From the figure, we can see that our model can automatically find suitable representations for different object categories. For object categories with stable 2D shapes, the model automatically select 2D primitives to form the rough shape of a category. For some portions of these objects, such as the top of tomatoes, handle of cups and base of the desktop globe, the algorithm will select 3D primitives, because these details are view specific and only appear in part of the view sphere.

For categories with complex shapes, coding the general shape for each view will be less efficient than coding the general 3D shape using 3D primitives. So, the algorithm automatically transit to select 3D primitives. With these experiment results, one can see that our model can learn templates of different mixing proportions, according to the shape and appearance of different object categories. Therefore, we propose that different object categories should be represented differently, and the information gain serves as an numerical answer to this representation problem.

6. Conclusion and Future Work

This paper presents a model mixing both representations via 3D and 2D primitives, and a method learning mixed templates for view invariant object recognition. The model can automatically transit between and mix object-centered and viewer-centered representations, which could provide a numerical answer to justify the debate over the two types of representation.

In the future, we will incorporate part level templates as an intermediate level of our current representation, and learn hierarchical models to better explain object categorical images.

Acknowledgement: We would like to thank Dr. Haifeng Gong and Professor Ying Nian Wu's helpful discussion. This work is supported by NSF grant IIS-0713652, ONR grant N00014-07-M-0287. We would like to thank Lotus Hill Institute (LHI) for preparing dataset. Work at LHI is supported by China 863 project 2009AA01Z331, 2008AA01Z126 and NSFC grant 60832004.

References

[1] I. Biederman and P. C. Gerhardstein. Recognizing depth-rotated objects: Evidence and conditions for three-dimensional viewpoint invariance. *Experimental Psychology: Human Perception and Performance*, 1993.

[2] I. Biederman and P. C. Gerhardstein. Viewpoint-dependent mechanisms in visual object recognition: Reply to tarr and bülthoff (1995). *Journal of Experimental Psychology: Human Perception and Performance*, 1995.

[3] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[4] M. Brown and D. Lowe. Unsupervised 3d object recognition and reconstruction in unordered datasets. In *3DIM*, 2005.

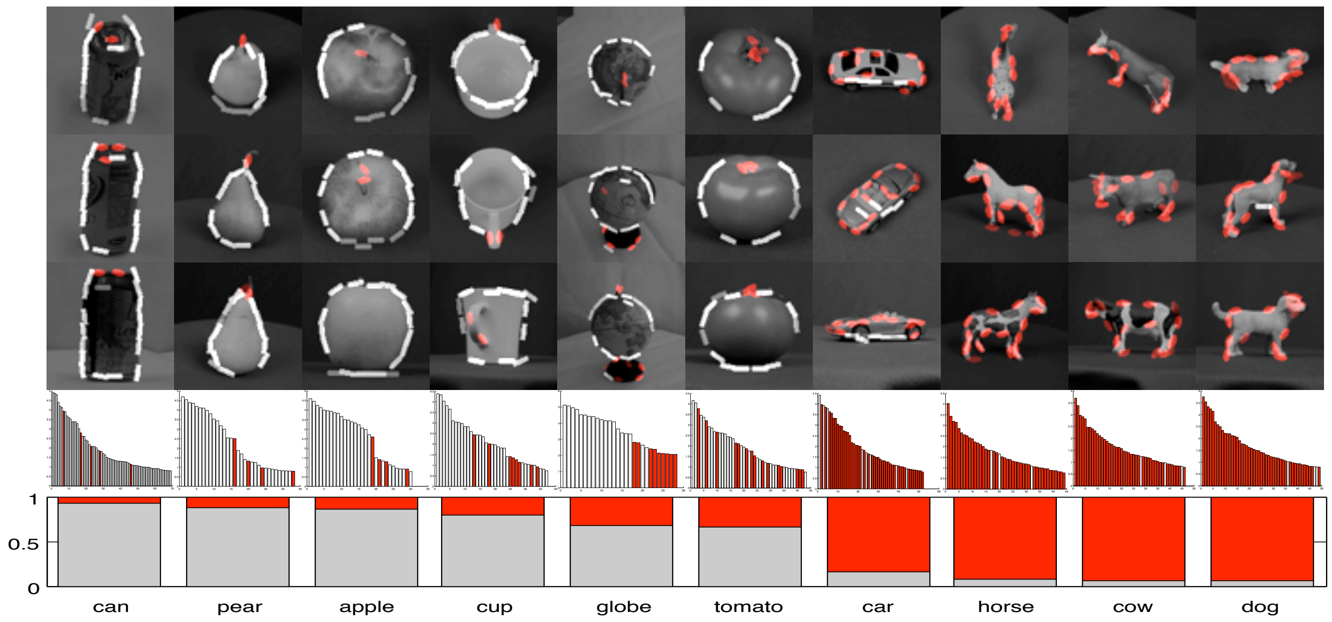


Figure 8. Learned object templates for 10 object categories and their pursuit indexes (Better viewed in color). **Row.1-Row.3**: Learned templates for different object images. Gray bars Represent 2D primitives and red ellipses are 3D primitives, intensity of those symbols show the corresponding score of each primitive. **Row.4**: The primitive pursuit index of each object category. The horizontal axis represents the pursuit order of each primitive, and vertical axis represents the information gain of each primitive. Again, red for 3D primitives and gray for 2D primitives. **Row.5**: The ratio on the number of 3D primitive (red) over 2D primitive (gray), for each category.

[5] W. G. Hayward and M. J. Tarr. Testing conditions for viewpoint invariance in object recognition. *Journal of Experimental Psychology: Human Perception and Performance*, 1997.

[6] D. Hoiem, C. Rother, and J. Winn. 3d layout crf for multi-view object class recognition and segmentation. In *CVPR*, 2007.

[7] E. T. Jaynes. Information theory and statistical mechanics. *Phys. Rev.*, 106(4):620–630, 1957.

[8] J. Koenderink and A. van Doorn. The internal representation of solid shape with respect to vision. *Biol. Cybern.*, 1979.

[9] A. Kushal and J. Ponce. Modeling 3d objects from stereo views and recognizing them in photographs. In *ECCV*, 2006.

[10] A. Kushal, C. Schmid, and J. Ponce. Flexible object models for category-level 3d object recognition. In *CVPR*, 2007.

[11] B. Leibe and B. Schiele. Analyzing appearance and contour based methods for object categorization. In *CVPR*, 2003.

[12] P. Leopardi. A partition of the unit sphere into regions of equal area and small diameter. *Electronic Transactions on Numerical Analysis*, 2006.

[13] J. Liebelt, C. Schmid, and K. Schertler. Viewpoint-independent object class detection using 3d feature maps. In *CVPR*, 2008.

[14] D. Marr. *Vision: a computational investigation into the human representation and processing of visual information*. W. H. Freeman, San Francisco, 1982.

[15] M. Minsky. A framework for representing knowledge. Technical report, Massachusetts Institute of Technology, 1974.

[16] S. D. Pietra, V. D. Pietra, and J. Lafferty. Inducing features of random fields. *TPAMI*, 1997.

[17] S. Savarese and L. Fei-Fei. 3d generic object categorization, localization and pose estimation. In *ICCV*, 2007.

[18] Z. Si, H. Gong, Y. N. Wu, and S.-C. Zhu. Learning mixed image templates for object recognition. In *CVPR*, 2009.

[19] S. Soatto. Actionable information in vision. In *ICCV*, 2009.

[20] H. Su, M. Sun, F.-F. Li, and S. Savarese. Learning a dense multi-view representation for detection, viewpoint classification and synthesis of object categories. In *ICCV*, 2009.

[21] M. Sun, H. Su, S. Savarese, and F.-F. Li. A multi-view probabilistic model for 3d object classes. In *CVPR*, 2009.

[22] M. J. Tarr and H. H. Bülthoff. Is human object recognition better described by geon structural descriptions or by multiple views? comment on biederman and gerhardstein (1993). *Journal of Experimental Psychology: Human Perception and Performance*, 1995.

[23] A. Thomas, V. Ferrar, B. Leibe, T. Tuytelaars, B. Schiel, and L. Van Gool. Towards multi-view object class detection. In *CVPR*, 2006.

[24] A. Torralba, K. P. Murphy, and W. T. Freeman. Sharing visual features for multiclass and multiview object detection. *TPAMI*, 2007.

[25] Y. N. Wu, Z. Si, H. Gong, and S.-C. Zhu. Learning active basis model for object detection and recognition. *IJCV*, 2009.

[26] P. Yan, S. Khan, and M. Shah. 3d model based object class detection in an arbitrary view. In *ICCV*, 2007.

[27] B. Yao, X. Yang, and S. C. Zhu. Introduction to a large scale general purpose ground truth dataset: methodology, annotation tool, and benchmarks. *EMMCVPR*, 2007.