Active View Selection for Object and Pose Recognition

Zhaoyin Jia, Yao-Jen Chang
Carnegie Mellon University
Pittsburgh, PA, USA
{zhaoyin, kevinchang}@cmu.edu

Tsuhan Chen
Cornell University
Ithaca, NY, USA
tsuhan@ece.cornell.edu

Abstract

In this paper we present an algorithm for multi-view object and pose recognition. In contrast to the existing work that focuses on modeling the object using the images only; we exploit the information on the image sequences and their relative 3D positions, because under many circumstances the movements between multi-views are accessible and can be controlled by the users. Thus we can calculate the next optimal place to take a picture based on previous behaviors, and perform the object/pose recognition based on these obtained images.

The proposed method uses HOG (Histograms of Oriented Gradient) and SVM (Support Vector Machine) as the basic object/pose classifier. To learn the optimal action, this algorithm makes use of a boosting method to find the best sequence across the multi-views. Then it exploits the relation between the different view points using the Adaboost algorithm. The experiment shows that the learned sequence improves recognition performance in early steps compared to a randomly selected sequence, and the proposed algorithm can achieve a better recognition accuracy than the baseline method.

1. Introduction

Object recognition is a key step in achieving the ultimate goal of artificial intelligence, and becomes one of the areas that has been extensively studied. With the development of machine learning algorithms and the appearance of advanced vision features, the performance of the object detection and recognition as well as pose estimation have been greatly improved [3, 4, 6, 10, 11]. However, most of this task have been done in detecting the objects in a single image. Given multiple images of an object, researches have been focused on the relation between the multi-views [14] or on the feature distributions in multiple images [7], such as modeling a panorama view of the object, and performing recognition on each image with information from different views. In addition, some studies explored models for 3D localization and pose estimation across multi-view images. In [8, 13], not only similar features in multiple images are considered helpful, the 3D relationship between the multi-view or multiple parts of one view are also taken into consideration. The multi-view models are linked by homography or affine transform, and therefore could be better aligned, resulting in an improvement in object recognition performance. With this 3D relationship, poses of the objects in the image can be also identified. Moreover, some work [15, 16] builds a full 3D model on multiple images using Structure from Motion method and relates testing images to this well-constructed 3D model for object recognition.

(a) initial view (b) turning 90^o (c) turning 180^o

Figure 1. A typical object and pose recognition task: (a) it is difficult to recognize the object and pose given the back view of a toaster; (b)(c) as more distinct views are acquired, object and all poses could be identified.

However, there are more relations to make use of within object recognition. Imagine a robot wandering in a room, trying to recognize specific objects by continuously taking pictures - a standard problem in multi-view object recognition. We may have access to the image sequence containing multiple pictures of the same object, which serves as the basic information for a typical multi-view recognition task. Furthermore, the relative 3D positions and the actions of the robot are also available to the user. Having the 3D pose of the robot to the object at each image could certainly help us in object/pose recognition. Once we gain a high belief in classifying one image into a category and a pose, this image could be used as a reference, and the remaining undetermined images could be recognized by using the relative 3D pose of the robot to the reference image. For instance, the back view of a toaster is difficult to recognize even for human. But after moving 180^o around the object and see-

ing the front view, both the object and the pose can be easily identified. From this front-view image, we can deduce that the previous image was the back of a toaster, as shown in Fig.1

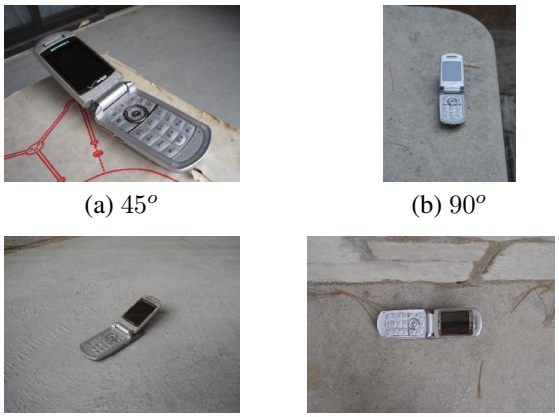
In addition, human beings are, in fact, conducting an active search during the recognition process. After seeing an object from one view, we are unlikely to take the following images sequentially with a constant degree interval such as $45^o, 90^o, 135^o \ldots$, but are more likely to choose the next action based on the previous recognition results. After combining all of the views obtained in this way, we can make a better decision in both category and pose classification. Inspired by such process, our work is aimed to simulate this behavior by finding the optimal sequence and linking the recognition results of different views.

There is some related work in this area. One application is the Curious George [9], a robot that explores its environment actively while identifying objects. Although the 3D environment is attentively searched in this work, the images are taken with fixed degree intervals and there is no feedback during the object recognition. In another study [1], the authors introduced a measurement of similarity across the multiple images of one object, and selected the most dissimilar view in similarity and scale space to perform recognition. However the definition of similarity seemed limiting for object recognition, which could not ensure the selection of the optimal sequence. Therefore, with advanced machine learning algorithms, [12] used the reinforcement Q-learning technique to train optimal action for object recognition. But this approach requires an intensive training process with a special device to perform the learning, which may not be easily transferred to the robot recognition task.

In this paper, we introduce an active search algorithm for multi-view object and pose recognition. The problem can be divided into two phases. The first phase is object/pose recognition in a single image, which is a typical problem in computer vision. The second phase is actions training based on the previous views, and the recognition linking from different views. We use Histograms of Oriented Gradient and Support Vector Machine to model this single image recognition problem [3], and use Adaboost to learn optimal actions. The paper will introduce the two steps sequentially. The first phase will be briefly described since it is a typical problem, while the the second phase will be discussed in more detail. Finally, experiments are conducted to evaluate its performance by using a 3D object dataset.

2. Dataset

One platform for active search in multi-view recognition would be a robot with controls to localize itself and a camera to take images within the unknown environment. To focus on the vision algorithm and make the results easily comparable, we use the UIUC Dataset of 3D object categories database [13] to simulate the behavior of the robot. In this dataset, multiple views of an object and the view angles are available as partially shown in Fig.2. For each object there are eight images with 45 degree intervals, and two to three variations in height and scale. The action of 'rotating the object for 45^o' in 3D space could be easily achieved by accessing the next image in the object image sequence.



(a) 45^o (b) 90^o (c) 135^o (d) 180^o

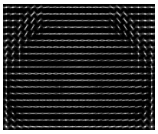
Figure 2. Partial images of the UIUC 3D Object Dataset which contains multiple views of one object at different view angles.

3. Recognition in a single image

We consider the multi-view recognition problem as a combination of several single-view recognition tasks. We build a model for each object at a certain angle, and the final multi-view recognition is done by linking the result of different single views. We use HOG [3] and SVM as the basic feature and classifier for single image classification, similar to the procedures conducted in [4]. HOG feature is useful in object recognition because such histograms largely maintain the shape of the objects and they are robust to some deformations of the objects in an image. However, with the UIUC 3D Dataset, different view points still impose large impact on the HOG feature. For example, a car from the back view and the side view are so different (shown in Fig.3) that if modeled together, the performance of the classifier could degrade dramatically.

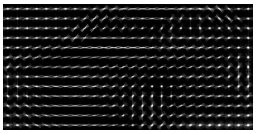
Therefore, we model an object at different views as a separate category. Although the complexity in learning and testing increases linearly to the number of the views, the recognition performance is greatly enhanced. Furthermore, modeling different views of the object will enable us to identify the pose of the object in the given image. We have eight models for one object and train the one vs. all Support Vector Machine [2] for object and pose classification, i.e. for each object at one specific view, we use samples with the same constraints as the positives, and samples of dif-

(a.1) back view of a car



(a.2) HOG feature

(b.1) side view of a car



(b.2) HOG feature

Figure 3. Although HOG can handle some deformations, images taken from different view angles of the same object exhibit quite different HOG features.

ferent objects, or the same object at different views as the negatives, to build a binary classifier.

4. Learning of the optimal action sequence

Suppose we have m different categories of objects with n different views for each object. After the previous step we would have mn different SVM models all together, that is, mn hypotheses for each testing image. Given one testing image, only one among mn hypotheses is correct. Our framework is that we can control the robot to rotate a certain degree and get a new image to gain more information to verify the proposed hypothesis. Therefore, the problem switches to whether we can determine a set of efficient actions so that the true hypothesis would gain more beliefs, and the other $mn - 1$ false hypotheses would be rejected.

The idea of active search is trying to learn the optimal sequence given the previous images. If we consider the SVM result on each view as one classifier, we can combine the information from multi-views by assembling the SVM results of those different views. It is identical to combining various weak classifiers to form a strong classifier, which is the basic idea of boosting algorithm in machine learning. We choose to use the Adaboost algorithm to achieve this combination as well as the optimal sequence. Adaboost [5] is a machine learning algorithm that minimize the exponential loss error. Similar to the other boosting algorithms, it exhaustively searches the optimal weak classifier, usually a decision stump, for classification and combines weighted weak classifiers to form a strong classifer.

4.1. Training instance collection

In the following sections, for training and testing, one instance means a complete image sequence that contains n different views of one object. With fixed recognition models and classifiers, the action to take in the next step should be based on the previous actions and images that have already been acquired. In other words, the next image needed should be the image of a view angle that could most disambiguate the category/pose confusion from the previous images. From this principle, an optimal sequence can be obtained. To achieve that, we collect a set of training instances for learning actions only. After we perform the boosting algorithm on this training set, we can get a learned image sequence representing an optimal set of actions based on the previous images.

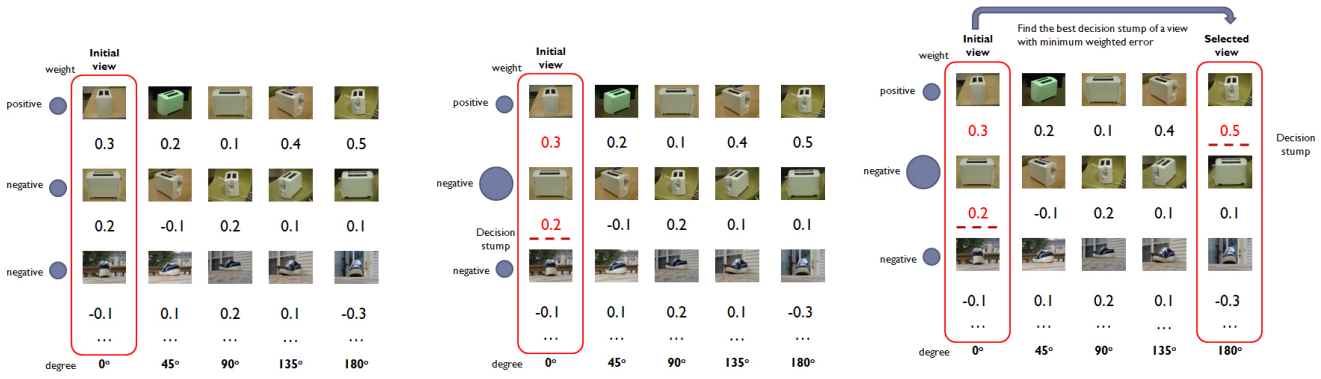
To simplify the problem, we focus on the case that for each instance, only one initial image is given. Each HOG/SVM is evaluated on this image, and then mn hypotheses are obtained, where only one hypothesis k is true. The next step has $n - 1$ choices. To form the positive instance, we put the following $n-1$ in the right order (increasing in every 45^o) relative to the true hypothesis k. And the negative instances are the ones with the wrong initial images, either with the wrong object or the wrong initial view. However, the sequence following the initial image should be in the correct order, whether this instance is true positive or true negative, so that 'move to the next image' truly corresponds to the robot 'moving to the 45^o view point relative to its current view point'.

For example, suppose we want to learn what actions to take when the given image is the back view of a toaster, denoted as $t(0^o)$. The positive instance would be $t(0^o), t(45^o), t(90^o), t(135^o), \ldots$. And the negative instance would be of the same category, but the initial view is wrong, such as $t(45^o), t(90^o), t(135^o), t(180^o), \ldots$, or the wrong category such as that of a car: $c(0^o), c(45^o), c(90^o), c(135^o), \ldots$. We use all the positives in the dataset and randomly choose twice as many negatives, and run the boosting algorithm on these instances.

4.2. Action sequence learning

4.2.1 Weak classifier

The boosting algorithm uses a combination of weak classifiers to form a strong one. In this paper we use the positive and negative decision stump of the SVM response on each image as the weak classifier. More specifically, we can have all the positives and negatives instances from the previous section piled into one $M \times n$ matrix, where M is the number of positive and negative training instance, and n is the number of different views. Each row represents a different training instance, and each column means a 'moving 45^o' action relative to its left column. Remembering each instance includes a set of images representing difference view angles, one image could be considered as a feature in identifying the initial hypothesis. We have mn hypothesis, and therefore mn SVM responses for one image, which is the belief margin of each hypothesis on this image. We make a positive or negative decision stump t on these SVM responses.



(a) (b) (c)

Figure 4. (a) Pile all of the training instance (each row) into one image matrix. Initially each instance is equally weighted. (b) Choose the best decision stump (red dashed line) with minimal training error on the initial view. Then update the weight on every instance. (c) Choose the next optimal view that minimizes the weighted error. In this way the best action is learned.

Suppose we are trying to evaluate one hypothesis k for positive decision stump t. If the SVM response of hypothesis k on this image is greater than t, then we classify the instance as true for hypothesis k, otherwise it is false. Note that now each decision stump is performed only on one image in an instance, and an instance includes n different images.

4.2.2 Adaboost training

In the problem setting, the initial view is given without knowing its view point beforehand. Therefore, the task is to learn the best view based on the given initial view. Suppose we have M training instance $i = 1, 2, \ldots, M$, for each training instance i we have a weight w_i, and the weight is initialized identical for all of the instances, such as $w_i = 1/M$. First we evaluate all the possible decision stump of the SVM response in the current view (refer to the 1st column in Fig.4(a)), and choose the best one with the minimal training error (dashed line). We can think it is the j-th step where $j = 1$. Once a decision stump on one image is selected on the j-th step, we can calculate the weighted training error ε_j:

$$\varepsilon_j = \sum_{i=1}^{M} w_i [y_i \neq h_j(s_{ij})] \tag{1}$$

where y_i is the ground truth of the current hypothesis k, it is $+1$ if current hypothesis is correct for the image sequence, and -1 otherwise. We use x_{ij} to denote the feature of image corresponding to j-th step in the i-th instance (i-th image sequence), and s_{ij} as the SVM response of the feature x_{ij}. Then $h_j(s_{ij})$ is the prediction of the decision stump t on response s_{ij} of the image x_{ij}. We have:

$$h_j(s_{ij}) = \begin{cases} 1, s_{ij} > t \\ -1, s_{ij} \leq t \end{cases} \tag{2}$$

where 1 and -1 means positive and negative prediction, respectively. Afterwards, we can compute the weight α_j for current decision stump h_j:

$$\alpha_j = 0.5 \ln \frac{1 - \varepsilon_j}{\varepsilon_j} \tag{3}$$

And the weight update is a key step to shift the focus of the combined classifiers to those 'hard instance' in the training process. The rule for updating the weight w_i of each instance for the next step is:

$$w_i^{(j+1)} = w_i^{(j)} e^{-y_i \alpha_i h_j(s_{ij})} / Z_j \tag{4}$$

where Z_j is the normalized constant to ensure $\sum_{i=1}^{M} w_i^{j+1} = 1$. All of the previous updating methods are visualized in Fig.4(b).

After updating the weight, we are able to select the next optimal view. We again evaluate every the possible decision stump on all of views, and choose the best one that minimizes the weighted error. The view contains the best decision stump becomes the next optimal view based on the $(j+1)$-th view, as shown in Fig.4(c). Then we can update α_j and weight w_i for each instance, and select the $(j+2)$-th optimal view iteratively. After obtaining all the views, the Adaboost algorithm can give a set of actions, i.e., the learned image sequence, that could optimally verify the hypothesis k, and in this way the optimal actions of hypothesis k is learned as shown in Fig.4. In Fig.5, an actual learned optimal sequence after boosting algorithm is shown based on a given initial image of an iron.

4.3. Testing

After the training step, for each hypothesis we have a classifier h, which includes a set of weak classifiers $\{h_j\}$ and their weights $\{\alpha_j\}$. The testing could be done as

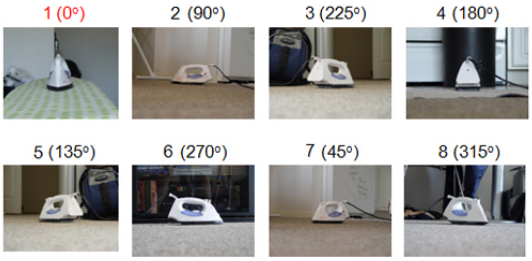


Figure 5. One sequence (sequence index and view angle) learned after given the initial view angle 0^o (index no.1, in red).

follows: For one hypothesis (*e.g.* the given image sequence $\{x_{ij}\}$ should be $x_{i1} = t(0^o), x_{i2} = t(45^o), x_{i3} = t(90^o), \ldots$) and its classifier h, evaluate whether it is true $(+1)$ or false (-1) for the given image. So given an initial image along with a hypothesis, each h performs as a look-up table, which composes of a sequence of learned viewing angles. And h_j is identical to the positions it should move to at step j. Thus h represents the action that should be taken under current hypothesis. During the testing we follow the action from h_j, which tells us where the next image should be taken. And the combined classifier is:

$$h(x_i) = \sum_j \alpha_j h_j(s_{ij}) \tag{5}$$

where x_i represents one testing image sequence as $\{x_{ij}\}, j = 1, 2, \ldots, n$. And the same s_{ij} is the SVM response of the HOG feature in image x_{ij}.

Note that the weak classifier is a simple decision stump of SVM response on mn hypotheses, it is possible to have multiple hypotheses become true for one image. To enforce the constraint that only one hypothesis can be true, we choose the one with maximum SVM response as the predicted hypothesis.

It is also possible to occur that the current prediction of the initial hypothesis to be false. There could be two strategies in the process of the active search to resolve the contradiction between the current predicted hypothesis and the initial hypothesis, as shown in Fig.6. In this situation, one can start a new hypothesis that has the maximum belief based on the images obtained so far. And thus it forms a one vs. another classifier, which gives an exact prediction of the given image sequence. Another strategy is to continue verifying the initial hypothesis, i.e. continue to follow the steps in current classifier $h = \{h_j\}$. Then h becomes a look-up table for the actions and the classifier perform a one vs. all binary classifier, which tells whether the initial hypothesis is true or false. In this paper we use the second strategy.

5. Experimental results

We use nine categories (car, toaster, cell phone, mouse, bicycle, iron, stapler, shoe, head) in the UIUC 3D object

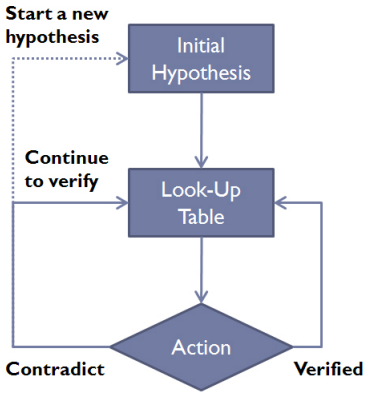


Figure 6. There are two strategies during the active search when the current predicted hypothesis contradicts with the initial hypothesis. Currently we continue to verify the initial one and the final output is a one vs. all classifier.

dataset, and for each object there are eight different views. We use roughly 2000 images to train the SVM and HOG model for different categories, 2000 images as the positives to train the actions, and 3000 images as the positives for testing. During the testing phase we randomly choose twice as many negatives with wrong category or initial view. We try to identify the pose and the category of the testing instances. For a single image during the testing, the SVM response is evaluated from the learned model with the HOG feature of this image.

A baseline of random action is implemented for comparison: we randomly select the views, representing the random actions and forming an image sequence $\{x_{ij}\}$. For each step j, we classify current image by the trained SVM classifier, and record the SVM responses s_{ij}. We choose the maximum SVM response belief as the predicted hypothesis, i.e. $max(s_{ij}), j = 1, 2, \ldots, j_{current}$, which is widely adopted in robot and recognition applications [9].

In this paper, two contributions are made: optimal action selection and the linking of recognition results from different views by Adaboost. To make a proper evaluation on these two contributions, we conduct three groups of experiments: (1) the baseline recognition method with randomly selected image sequence (Red line in Figs.7 and 8); (2) the baseline recognition method with the learned optimal image sequence (Green line); and (3) the proposed Adaboost algorithm for multi-view recognition with the learned sequence (Blue line).

We perform a one-vs-all classification in all of the three experiments, which means for an input image sequence $\{x_{ij}\}$, we evaluate one hypothesis on it, and the result would be whether this hypothesis is true $(+1)$ or not (-1). Therefore, the random chance would be 50%.

Fig.7 shows the average category and pose recognition accuracy for the three experiments over nine categories.

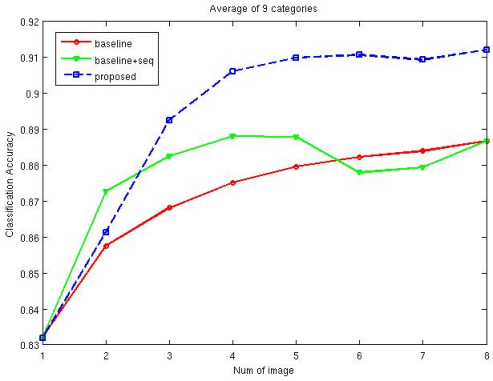


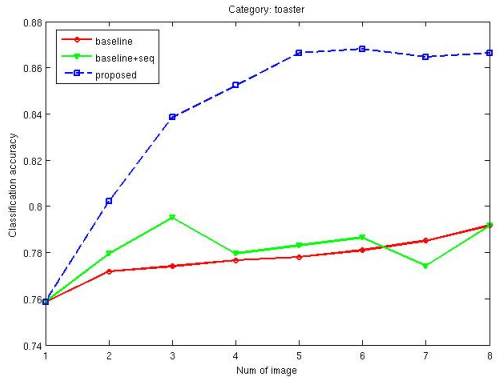
Figure 7. The average recognition accuracy of object and pose, including three different methods.

The performance of these three methods starts at the same level in the initial step since the same initial image is given. Comparing the red line(baseline recognition and the randomly selected sequence) and the green line(baseline recognition and learned sequence), one can find that the learned sequence gives improvements in the first few steps, because it selects the optimal views based on the given initial image, i.e., choose a better angle to take the next picture, which results in disambiguating the object/pose. In Fig.5, the initial given view of an iron is the front view, which is hard to identify; but the learned sequence moves to the side view of the iron, and thus results in a better recognition accuracy.

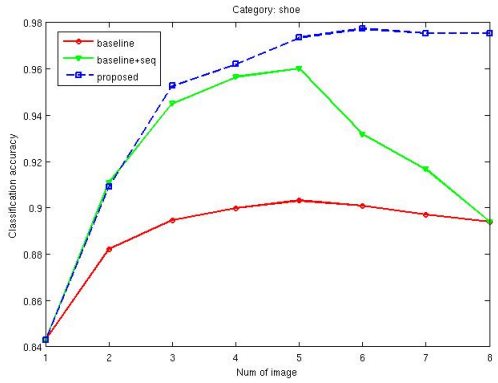
However, when we continue to acquire more images, the performance of green line drops, which means the baseline recognition method become confused with the learned sequence after some steps. It is mainly because some quite confusing views are now obtained, such as the two side views of a car, which may lead to a misclassification with very high belief. Finally, the green line and red line meet when all the images in the sequence are obtained, because the recognition technique is the same, and after acquiring all the images they should result in the same accuracy.

The proposed Adaboost algorithm (blue line) have a significant boost in the process, and also results in a better performance when more images have been acquired. One advantage of the proposed method is that it learns the optimal action to take based on the previous results, so the recognition accuracy is increased more rapidly than randomly chosen actions. Another advantage is that during the boosting process, more information from the previous images is passed through the combination process. Assisted with the learned weights α_i, the confusing angles are weighted less than the former confident ones, thus a better performance can be maintained even some confusing views are acquired.

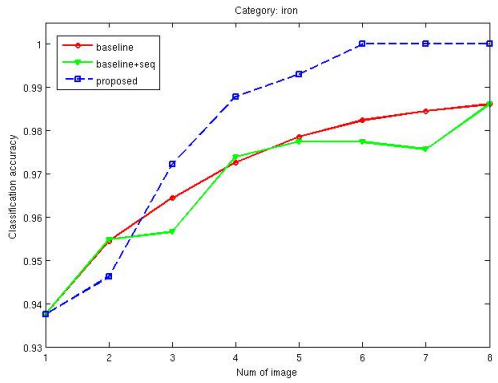
In Fig.8, we take a close look on experiment results of different categories including three categories: 'toaster', 'shoe', and 'iron'. For the first two categories, shown in



(a)



(b)



(c)

Figure 8. Testing accuracy on three different categories: (a) toaster, (b) shoe, and (c) iron.

Fig.8(a) and (b), the learned sequence gives a clear enhancement to the recognition performance compared to the random sequence, and also the proposed algorithm outperforms the baseline algorithms. However, there is a case that the learned sequence cannot give a significant boost and the improvement is not stable, such as the category 'iron' shown in Fig8(c).

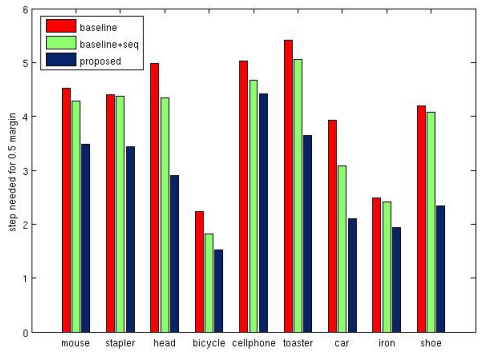


Figure 9. The average steps required to achieve 0.5 belief margin.

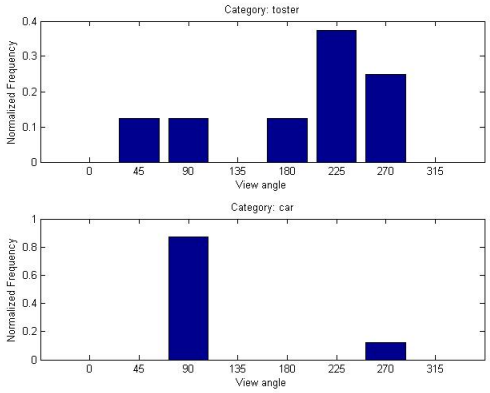


Figure 10. The histograms of the second selected view for categories toaster (above) and car (below). Histograms have been normalized to sum to 1

One reason to explain this fact is that, the learned sequence is generated by the boosting algorithm, but recognition method is using the baseline, thus this combined one lacks the weighting process from the Adaboost algorithm. The learned sequence in this case only gives a better view selection. Although in many cases (such as (a) and (b) in Fig.8) the learned sequence gives an enhancement to the performance, it is not guaranteed, for it cannot weight the confusing views less to gain a higher and correct belief. However the proposed algorithm still outperforms the baseline when acquiring more images.

The evaluation of how fast the active search algorithm can identify the category and pose of the object is also conducted. We calculate the margin of the classification at each step, and stop acquiring new image if the belief margin exceeds 0.5. In Fig.9 the result shows that the proposed Adaboost algorithm uses fewer steps to achieve this belief margin (blue bar), and the learned sequence (green bar) has a better performance compared to the baseline (red bar) even with the baseline recognition method. Therefore, it is more efficient to use the learned action sequence for recognition.

Another interesting issue is what absolute view angles are taken by the algorithm. We present the histograms of the second selected view for two categories, car and toaster, in Fig.10. With different initial steps, for category 'toaster' the picked second views are more uniformly distributed. In contrast, side views are preferred for the category 'car'. The reason is that comparing to the 'squarish' toaster, car has a more distinct view, and the algorithm is picking up these views at early steps regardless of the initial view. This also explains that a car requires fewer steps to identify than a toaster, as shown in Fig.9.

6. Conclusions

In this paper, we presented an algorithm to achieve active search in multi-view object and pose recognition problem. We use SVM with HOG features to build the basic classifier, and then propose an algorithm to learn the optimal action for recognition by using a boosting method. Result shows that our method has a better performance than randomly selected sequence in recognition accuracy. It is also more efficient that achieves high performance with fewer images.

One future work of the active search would be boosting in features. Currently the extra information gained across the multiple images is only limited to the responses of single classifiers. However, it is possible to build a part model of the object, which will enable us to pick up the most distinct part across the multi-views, and the recognition performance could be enhanced by acquiring new parts. Another extension of this work would be testing on real applications. When implemented on a robot or camera array, the active search space could be extended into scale and height besides the view angle, and the space is denser rather than sparse degree intervals such as 45^o conducted in this work. Combining tracking with recognition could also be an interesting direction for further investigation.

References

[1] S. Abbasi and F. Mokhtarian. Automatic view selection in multi-view object recognition. In *ICPR*, pages Vol I: 13–16, 2000.

[2] S. Canu, Y. Grandvalet, V. Guigue, and A. Rakotomamonjy. Svm and kernel methods matlab toolbox. Perception Systemes et Information, INSA de Rouen, Rouen, France, 2005.

[3] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, pages I: 886–893, 2005.

[4] P. F. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *CVPR*, pages 1–8, 2008.

[5] Freund and Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *JCSS: Journal of Computer and System Sciences*, 55, 1997.

[6] D. G. Lowe. Object recognition from local scale-invariant features. In *ICCV*, pages 1150–1157, 1999.

[7] D. G. Lowe. Local feature view clustering for 3D object recognition. In *CVPR*, pages 682–688. IEEE Computer Society, 2001.

[8] S. S. M. Sun, H. Su and F.F.Li. A multi-view probabilistic model for 3d object classes. In *CVPR*, pages 1–8, 2009.

[9] D. Meger, P.-E. Forssén, K. Lai, S. Helmer, S. McCann, T. Southey, M. A. Baumann, J. J. Little, and D. G. Lowe. Curious george: An attentive semantic robot. *Robotics and Autonomous Systems*, 56(6):503–511, 2008.

[10] V. M.Ozuysal and P.Fua. Pose estimation for category specific multiview object localization. In *CVPR*, pages 1–8, 2009.

[11] K. P. Murphy, A. Torralba, D. Eaton, and W. T. Freeman. Object detection and localization using local and global features. In *Toward Category-Level Object Recognition*, pages 382–400, 2006.

[12] L. Paletta and A. Pinz. Active object recognition by view integration and reinforcement learning. *Robotics and Autonomous Systems*, 31(1-2):71–86, 2000.

[13] S. Savarese and F. F. Li. 3D generic object categorization, localization and pose estimation. In *ICCV*, pages 1–8, 2007.

[14] A. Thomas, V. Ferrar, B. Leibe, T. Tuytelaars, B. Schiel, and L. J. V. Gool. Towards multi-view object class detection. In *CVPR*, pages II: 1589–1596, 2006.

[15] J. X. Xiao, J. N. Chen, D. Y. Yeung, and L. Quan. Structuring visual words in 3D for arbitrary-view object localization. In *ECCV*, pages III: 725–737, 2008.

[16] P. K. Yan, S. M. Khan, and M. Shah. 3D model based object class detection in an arbitrary view. In *ICCV*, pages 1–6, 2007.