A Novel Rotation Adaptive Object Detection Method Based on Pair Hough Model

Yang Liu[a,], Lei Huang[a], Xianglong Liu[a], Bo Lang[a]

[a]*State Key Laboratory of Software Development Environment, Beihang University, China*

Abstract

This paper proposes a novel Hough-based object shape representation model called Pair Hough Model (PHM) and its corresponding object detection framework. PHM constructs the voting models implicitly with automatically detected interest points and their local descriptors for unseen object categories. In addition, by casting votes according to key point pairs instead of individual key points and taking the orientations of objects as well as their sizes into consideration, PHM can recognize and localize objects after their scaling and/or rotation, which makes it suitable for processing images with major rotations such as pictures taken by mobile devices. Evaluation experiments proved that PHM does not need to be trained on rotated images to recognize rotated objects, and PHM achieved comparable results to the state-of-the-art methods on several widely used public data sets.

Keywords: Object categorization, Object detection, Generalized Hough transform, Rotation Adaptive

1. Introduction

After years of research, current object recognition approaches are already capable of identifying a large range of different objects. For example, Generalized Hough Transform [1] has been widely used in object recognition and localization for its simplicity and efficiency, and its successors such as Implicit Shape Model (ISM)[2] and Class-specific Hough Forest (CHF) [3] have extended the use of Generalized Hough Transform to unseen object recognition. Deformable Part Model (DPM)[4] and Selective Search [5] are newly

Email address: `xlliu@nlsde.buaa.edu.cn` (Xianglong Liu)

proposed methods that emerged from Pascal Visual Object Classes (VOC) Challenge [6]. They both achieved impressive results on VOC data sets, and DPM has already become a popular base of other object recognition methods and also been extended to other territory such as scene recognition [7].

However, none of the above methods have any coping mechanism with rotation. As a matter of fact, the rotation of objects is mostly ignored by algorithm designers and data set builders. Although not a major problem in most data sets for object detection tasks, rotated objects, or more usually rotated views, can be found in many situations, e.g. Fig.1 shows a few rotated objects in VOC2007 data set. Some of the rotations occurring in pictures are created on purpose for some kind of artistic impression, and some of them are caused by the instability of camera support, especially when the pictures are taken by mobile devices such as smart phones [8, 9, 10]. Therefore, exploring the ways of recognizing rotated objects while not losing precision on standard object detection task is worthy for analyzing fast increasing user produced images in social media.

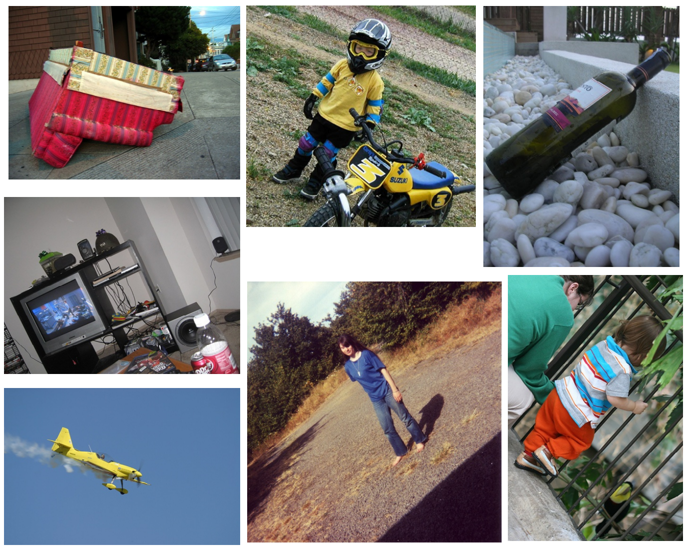


Figure 1: Rotated objects and views from PASCAL VOC2007 data set for object detection task.

In this paper, Pair Hough Model (PHM), which is based on Generalized Hough Transform and Mean-shift Model Estimation [11], is proposed as a novel object detection method focusing on handling rotated objects. PHM uses automatically detected interest points as voting subjects in order to handle the rotation of objects. So far, a wide variety of interest point detectors and local feature descriptors have been proposed, e.g. [12, 13, 14, 15, 16, 17, 18, 19]. Many of these detectors, including the Difference of Gaussian (DoG) detector [13] and the Fast-Hessian (FH) detector [14], are designed to tolerate in-plane rigid-body transformation like scaling and rotation, which makes them valid choices for our purpose.

When using individual key points to estimate the position, scale and orientation of the target object, the accuracy of vote casting depends greatly on the repeatability of interest point detection methods. However, the scale and orientation of key points usually cannot be detected accurately with the existing detectors. Meanwhile, when recognizing objects with smooth edges such as computer monitors and cars, interest points detected at corners (as in Fig.3) may not always reflect the scale of objects, e.g. when zooming in a rectangle, its edges extend accordingly, but the scale of a corner interest point does not usually increase proportionally to the size of the rectangle, depending on the detector. Therefore, instead of using individual key points as voting subjects, Pair Hough Model combines co-occurrent key points into pairs and takes advantage of their relative positions to obtain finer prediction of pose and size of the target object.

The **contribution of this paper** mainly lies on Pair Hough Model, which is a novel shape representing model for Generalized Hough Transform, and its corresponding object detection framework. Pair Hough Model achieved high robustness to in-plane rigid-body transformations by utilizing automatically detected key points and a key point pair based voting strategy.

The rest of this paper is structured as follows. Sec. 2 introduces some important related works, most of which will be compared with PHM in the experimental section. In Sec.3 and Sec.4, the Pair Hough Model and its corresponding recognition framework are presented in detail. Sec.5 gives the results of several experiments to evaluate the accuracy and transformation adaptivity of PHM. Then, a final discussion concludes our work in Sec. 6.

2. Related Works

The inspiration of Pair Hough Model is basically drawn from the three following methods: Generalized Hough Transform [1], Implicit Shape Model [2] and Class-specific Hough Forest [3].

Generalized Hough Transform [1] is a popular method in object recognition and localization for its simplicity and efficiency. In order to use Generalized Hough Transform to recognize objects, voting models of detectable parts need to be defined so that votes for object parameters such as position and size can be casted according to them. Most Hough-based face recognition techniques [20, 21, 22] use predefined models since frontal human faces share many common features. However, defining models for an unseen object category is difficult, and every object class is not so easy to be modeled as human faces.

To address this problem, Implicit Shape Model (ISM) [2], in which an object shape is defined implicitly as a collection of local features extracted from detected interest points, chooses to generate voting models automatically by gathering information from user annotations instead of defining voting models beforehand. Also by determining scale parameter with mean-shift search [11] instead of sliding window technique, ISM is fairly efficient when recognizing multi-scale objects.

Different from ISM, Class-specific Hough Forest (CHF) [3] uses dense sampling instead of interest point detection for voting subject finding. CHF builds a random forest of binary trees for sample quantization. Each leaf node in CHF corresponds to the training samples that reached this node during training, and each non-leaf node corresponds to a binary test, which separates samples reached this node into two sets while satisfying conditions given by the authors. CHF assumes that objects are in a same scale, so in order to process multi-scale images, CHF needs to scale the test images by a series of factors and then applies recognition method on the scaled images.

Besides the introduced Hough based methods, two newly proposed methods, which are Deformable Part Model [4] and Selective Search [5], are going to be used in experiments presented in Sec.5 for comparison.

Deformable Part Model (DPM) [4] uses Latent-SVM to train a seris of Histogram of Oriented Gradient (HoG) [23] models and their relationship with the object, while each of the HoG models represents an object part or the whole object. DPM showed very impressive results in the Pascal Visual Object Classes (VOC) Challenge [6] and has been used widely in object detection tasks.

Selective Search (SS) [5] is an alternative way of Exhaustive Search such as sliding window technique. In the algorithm proposed by Uijlings et al., selective search is done by applying pixel based segmentation methods on images and merging nearby regions to form possible object areas. Then, a time-consuming ranking step, which is based on densely sampled SIFT [13] (or SIFT-like) descriptors and Spatial Pyramid Matching [24], is employed to evaluate each object area given by the selective search step. This method has reported the best result in PASCAL VOC2012 object detection competition.

As has been mentioned in Sec.1, none of the above methods is capable of detecting rotated objects. Although the selective search step of SS can deal with rotated objects, its ranking step is highly sensitive to rotation.

The past years have seen a few works on rotated image recognition. Some of the researchers exploiting this territory focusing on transformation invariant features, e.g. [25] designed an invariant feature space by employing Gabor filters, [26] attempted to make HoG descriptors rotation invariant by applying a Fourier transform to their gradient histograms. Some others tried to modify existing pattern recognition algorithms, e.g. [27] increased the rotation adaptivity of SVM by transforming the original image into a Hough space and using a cross-correlation based kernel.

MICS [28] uses local feature pairs for spatial verification of digital document retrieval task and achieved good performance. In MICS, relative positions of feature pairs are used to solve the linear transformation matrix, and then a Hough transform are performed to verify the identicalness of two bag-of-word matched documents. PHM extends this idea into a more general form in order to detect arbitrarily located objects.

These works are important steps to real rotation invariant object recognition, but none of them is a complete object recognition process until [29] proposed a rotation invariant object detection method based on boosted random Ferns and HoG-based features. They decoupled the orientation estimation step from the classification step to avoid training different classifier for each possible orientation, i.e. first they determine the orientation of the target object with a pose estimator, then the original image is steered according to this estimated orientation and a classifier is applied to the rotated image. This method is fairly efficient due to this decoupling and has achieved high performance on two data sets of cars and motorbikes.

3. Pair Hough Model

Pair Hough Model (PHM) is the key of the object detection method proposed in this paper. PHM defines the object shape representation on the basis of automatically detected interest points and forms the foundation of recognition algorithms.

A PHM model is corresponding to a specific object category and composed of voting models, which indicate how detected key point pairs (KPPs) will cast votes in the voting process of Generalized Hough Transform. Therefore, the training process of PHM is basically building voting models for objects belonging to a same category. A brief process of PHM training is as follows:

1. **Feature Extraction**: Local features are extracted and quantized by a binary tree based coding system. The Code book construction and descriptor quantization will be presented in Sec. 3.1.
2. **KPP Construction**: Key point pairs are obtained by enumerating each two key points in an image and then normalized, as described in Sec. 3.2.
3. **Voting Model Building**: For each normalized KPP, a voting model is generated during the training. During the normalization of KPPs, the same transformation is applied to the space parameters (including center, size and orientation) of the target objects, and then the transformed object parameters are mapped into the model space of each KPP as vote points. Refer to Sec. 3.3 for more details.

3.1. Code Book Construction

As in most local feature based methods, the first task of PHM is to quantize each extracted feature so that features from different images can compared efficiently. In the experiments mentioned in Sec.5, binary coding trees instead of traditional linear code books are used for this task. This section will introduce the construction and use of coding trees in detail.

3.1.1. Construction and Use of Coding Trees

A coding tree used by PHM is practically a modified version of Class-specific Hough Forest (CHF) [3]. The most important difference between PHM and CHF is that PHM trains only one coding tree instead of a random forest for each object category. CHF builds a random forest of binary trees for sample quantization. During the training of each binary tree, samples are randomly chosen from the training images so that different trees may complement each other. In PHM, however, multiple coding trees will increase the run time of mean-shift iteration heavily, so only one coding tree is trained and all features extracted from training images are used for the training of each coding tree.

Similar to CHF, coding trees of PHM are binary trees. Each non-leaf node in a coding tree is assigned a binary test to divide features arriving at this node into two sets, and the binary tests are trained by Linear SVM. The detailed process of binary test training is presented in Sec.3.1.2. Every leaf node in a coding tree is given a unique code number for identification and a weight equal to the key points within the object areas divided by all key points that have reached this leaf node.

By passing the descriptor of an extracted key point down a trained coding tree, this key point will eventually reach a leaf node and be assigned the code number of that leaf node. Thus, a quantized key point can be represented by a 3-tuple $K_P = (c, s, a)$, where c is its code number, s is its scale and a is its orientation.

3.1.2. Binary Test Training

When training a non-leaf node, key points that have reached this node are divided into 3×16 sets according to their positions and scales relative to the target objects. First, in order to process key points from different images together, each object bound box will be normalized into a unit square and its corresponding key points are transformed along with it, i.e. relative position and scale of each key point will be calculated by using the following equations.

$$x_r = \frac{x - x_b}{w_b} \tag{1}$$

$$y_r = \frac{y - y_b}{h_b} \tag{2}$$

$$s_r = \frac{s}{\max\{w_b, h_b\}} \tag{3}$$

In Equ.(1)-(3), (x_r, y_r) is the relative position of the key point and s_r is the relative scale. (x_b, y_b) is the center point of the corresponding bound box and w_b/h_b is the width/height of the bound box.

Then, the normalized bound box is segmented by a 4×4 grid, and key points falling in each grid cell form a set. Key points which are outside the object area are assigned to their nearest grid cells. After that, key points falling in one grid cell will be divided further into 3 individual sets which correspond to scale range $[0, \frac{1}{3})$, $[\frac{1}{3}, \frac{2}{3})$ and $[\frac{2}{3}, 1)$ respectively.

After partitioning the key points, between each two key point sets a Linear SVM classifier is trained to classify key points based on to which set them most likely belong. Naturally, different classifiers will have different precision.

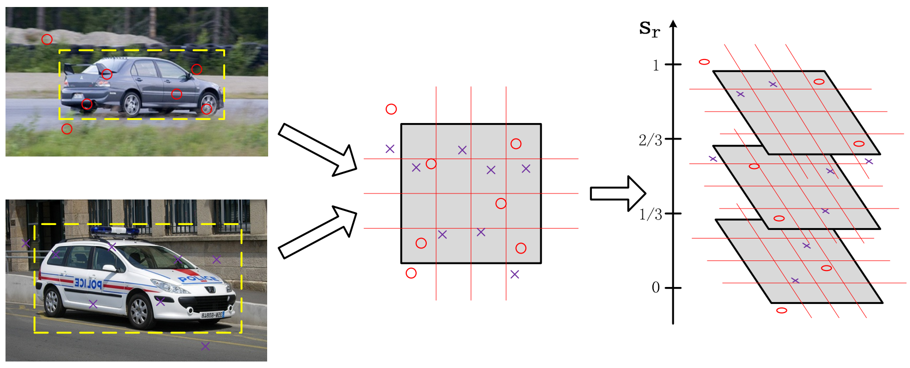


Figure 2: Demonstration of key point partitioning for binary test training. Circles and crosses represent key points from different samples.

The classifier with the highest precision is chosen to be the binary test of the current node.

The *Balanced Precision* (BP) is used for evaluating the precision of classifiers, because if the *Overall Precision* (correctly classified key points divided by all key points) is used to determine which classifier to choose, when two sets are extremely unbalanced, e.g. one set has 100 key points and the other one has 1 key point, a classifier which simply puts all samples into one class will also have a very high score. The Balanced Precision is calculated with the following equation.

$$BP = \frac{1}{2}\left(\frac{C_p}{T_p} + \frac{C_n}{T_n}\right). \tag{4}$$

In Equ.(4), T_p (T_n) is the total number of positive (negative) samples, and C_p (C_n) is the number of correctly classified positive (negative) samples.

Termination Criterion. Under some circumstances, a node should stop splitting in order to prevent the tree from growing overly large. In our experiments, when a tree node reaches a limited depth and/or has too few key points and/or all trained classifiers has a BP no more than 0.5, it is declared as leaf node and will not split further.

3.2. Key Point Pair Representation

Once interest points are detected and quantized, Key Point Pairs (KPPs) can be obtained by iterating over all pairs of interest points. Since PHM is supposed to handle transformations such as scale and rotation, the representation of KPPs need to be chosen carefully in order to provide efficient ways of comparing KPPs from different images.

3.2.1. Different types of KPPs

Relative to an object, there are 4 types of KPPs that are useful to object detection task, as in Fig.3.

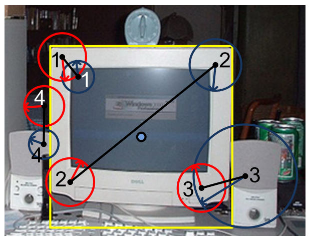


Figure 3: Different kinds of KPPs relative to the object. Circles represent key points. A KPP is given by a line segment connecting two key points. Numbers in the circles are used to label different type of KPPs.

KPPs of type 1 and 2 are within the area occupied by the object, so they are probably strongly related to the object. The two types are not completely the same. KPPs of type 1 are composed of two key points close to each other, so they mostly carry information about a small part of the object. By contrast, KPPs of type 2 may carry information of large scale structures of the object.

KPPs of type 3 and 4 are partly out of the annotated object area, i.e. at least one of the two key points has a neighborhood mostly belongs to the background, so their relationships with the object are weaker than those of the former two types. However, these KPPs may still have their value to the recognition task, because objects of a same category usually appear in similar environments, e.g. a monitor usually comes along with the other parts of the computer such as speakers, keyboard, mouse and so on.

3.2.2. Normalization of KPPs

Identical KPPs found in different images may still vary in scale and/or orientation, so KPPs will be normalized before comparison in order to achieve robustness to scaling and rotation. As in Fig.4, the normalization of KPPs is performed by applying to the image space a linear transformation that

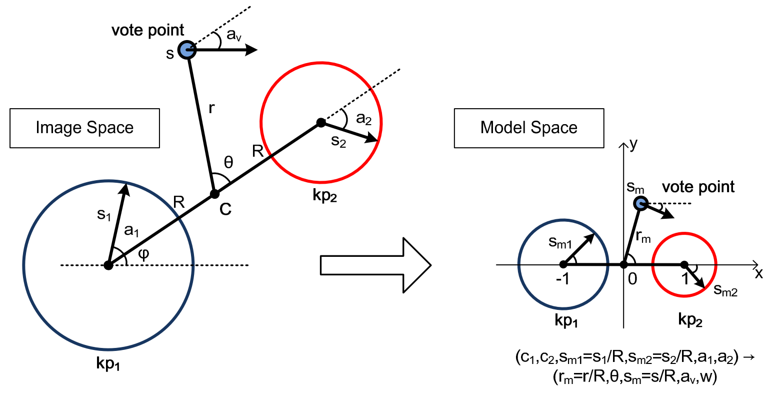


Figure 4: Transformation from image space to model space. In each half of the figure (left and right), the large circles represent two different key points with orientations shown by the arrows from their centers to boundaries. C in the left part is the middle point of the edge connecting kp_1 and kp_2, and the distance between C and each key point is R. The small circles represent vote points in both image space and model space with arrows giving their orientations, and r is the distance between the vote point and C in image space. φ is the angle between the horizontal line and the edge connecting kp_1 and kp_2.

places the two key points on $(1, 0)$ and $(-1, 0)$ respectively. In this way, the influence of both scaling and rotation of the image could be eliminated.

Since the order of two key points in an image is arbitrary, a sorting strategy needs to be defined in order to determine which key point should be placed on $(1, 0)$ during the normalization. In experiments, we sort key points by their code number, and when the two key points have one same code number, each order of the two key points is treated as an individual KPP.

A normalized KPP (kp'_1, kp'_2) in the model space is represented by a 6-tuple $P_M = (c_1, c_2, s_{m1}, s_{m2}, a_1, a_2)$, where c_i is the code number of kp_i, $s_{mi} = s_i/R$ is the normalized scale of kp_i with s_i being the scale of kp_i, a_i is the angle between the orientation of kp_i and the edge connecting the two key points, and R is half the distance between kp_1 and kp_2.

The following proposition states the fact that the normalization of KPPs can eliminate the influence of in-plane rigid-body transformations of the test images.

Proposition 3.1. *When an image is converted into another image by multiplying the coordinates of each of its pixel by a linear transformation matrix* $\mathbf{A}$ *in the form of Eqn(5), if the interest point detector and local descriptor* *are scale and rotation invariant, a KPP* (kp_1, kp_2) *in the original image and its corresponding KPP* (kp'_1, kp'_2) *in the transformed image will be assigned the same 6-tuple after the normalization.*

$$\mathbf{A} \;=\; \mathbf{TSR} = \begin{pmatrix} 1 & 0 & \Delta_1 \\ 0 & 1 & \Delta_2 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \gamma & 0 & 0 \\ 0 & \gamma & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\delta & -\sin\delta & 0 \\ \sin\delta & \cos\delta & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (5)$$

The proof of Prop.3.1 will be given in the Appendix.

However, there is no interest point detector that is ideally scale and rotation invariant, that makes slight deviations of key points unavoidable during a transformation. Therefore, a simple quantization is employed in this paper to **discretize** P_M: the space of P_M is divided into fixed-size grids and KPPs which fall in a same grid are considered identical in the following processes. In this way, deviations of key points can be tolerated in a certain degree depending on the chosen size of grids.

3.3. KPP Based Hough Voting

After KPPs are extracted, normalized and discretized as in Sec.3.2.2, voting models can be constructed with respect to each KPP, and then they can be used to cast votes during testing. In this section, the construction of PHM voting models and its corresponding voting strategy will be discussed in detail.

3.3.1. Construction of Voting Models

In PHM, a voting model is composed of many vote points each representing a possible position of the object, and it is corresponding to a specific KPP in the model space. For the reason stated in the end of Sec.3.2.2, KPPs which are identical after been discretized will share a voting model in the rest of the voting process.

An object in the image is corresponding to a vote point denoted by (x, y, s, a), where (x, y) is the center of this object, s is the size, and a is its orientation. During the training stage, each vote point will be mapped into the model space of each KPP by applying the linear transformation in Fig.4, i.e. if an image contains n KPPs, an object in that image will generate n vote points, each of which is corresponding to a different KPP and mapped into the model space of that KPP. Also, since two KPPs in the image space may corresponding to a same 6-tuple after been normalized and discretized, an object may generate multiple vote points in one model space.

Transformed vote points in the model space are represented with a 5-tuple $V_M = (r_m, \theta, s_m, a_v, w)$, in which $r_m = r/R$ and θ are the polar coordinates of the vote point in the model space, $s_m = s/R$ is the the normalized object size, a_v is the angle between the orientation of target object and the edge connecting two key points, and w is the weight of this vote point with expression as follows

$$w = \frac{w_1 \cdot w_2}{t} \tag{6}$$

In equation (6), t is the total count of appearances of this KPP in training images, and $w_1(w_2)$ is the weight of leaf node $c_1(c_2)$ in the coding tree.

The motivation of dividing weight w by t is that if a KPP frequently appears in the same position relative to the object, it may cast too many vote points in a small area during test and thus attract Mean-Shift iteration to this area even if there is no other pair voting for this position. Therefore, w of each vote point is divided by t to prevent a single KPP from being overly influent.

3.3.2. Voting Strategy

In the recognition stage, a KPP $P_I = (kp_1, kp_2)$ extracted from the test image is mapped into model space to find P_M, as in Fig.4. Then, P_I will cast votes according to each vote point in the voting model of P_M. The following equations are used to transform a model space vote point to an image space vote point.

$$\begin{aligned}
x_{vote} &= x_C + r_m{\cdot}R{\cdot}\cos(\theta + \varphi) && (7)\\
y_{vote} &= y_C + r_m{\cdot}R{\cdot}\sin(\theta + \varphi) && (8)\\
s_{vote} &= s_m{\cdot}R && (9)\\
a_{vote} &= a_v + \varphi && (10)
\end{aligned}$$

x_C and y_C are coordinates of the middle point C of P_I. r_m, s_m, φ and R are defined in Fig.4.

By using the transformation shown in Fig.4, PHM can easily maintain the position, size and orientation of objects when the image is scaled and/or rotated, as is stated in the following proposition.

Proposition 3.2. *When an image is converted into another image by applying the transformation described in Prop.3.1, and the key point detector and local descriptor are scale and rotation invariant, if voting models are trained in the original image, the number of votes casted at the transformed object position, scale and orientation will be at least the same as the number of KPPs in the original image during the testing on the transformed image.*

The Appendix will give the proof of Prop.3.2.

However, in practice, annotations are usually given in the form of bound boxes, so all the 4 parameters cannot be obtained accurately, especially the orientation since annotations in most data sets come without detailed information of object rotation. Therefore, we use 3D continuous voting space $V_{space} = \{(x, y, s)|x \in \mathbb{R}, y \in \mathbb{R}, s \in \mathbb{R}, s > 0\}$ for vote casting, and a_{vote} is discretized into 4 ranges, including $(-22.5^o, 22.5^o] \cup (157.5^o, 202.5^o]$, $(22.5^o, 67.5^o] \cup (202.5^o, 247.5^o]$, $(67.5^o, 112.5^o] \cup (247.5^o, 292.5^o]$ and $(112.5^o, 157.5^o] \cup (292.5^o, 337.5^o]$. The ranges of a_{vote} are used for labeling a series of voting spaces ($0^o/180^o$, $45^o/225^o$, $90^o/270^o$ and $135^o/315^o$), and a_{vote} determines to which voting space the corresponding vote is supposed to be casted, as shown in Fig.5.

Also, as is stated in the end of Sec.3.2.2, unideal scale and rotation invariance of interest point detectors may cause vote points to spread a small area instead of centralizing densely in the real position of the object, which could affect the accuracy of voting process. However, in the recognition framework resented in Sec. 4, the maxima in voting spaces are found by MSME, which is based on Kernel Density Estimator. Therefore, with appropriate bandwidth, Kernel Density Estimator can tolerate vote point deviation caused by either deformation of objects or instability of interest point detectors, so the absolute accuracy of voting is not necessary for high recognition precision. ISM and ISM-like methods proved this fact.

4. Rotation Adaptive Object Recognition Framework

The training process of PHM, which has been presented in Sec.3, is basically the construction of voting models. After voting models are obtained, a Generalized Hough Transform is conducted to recognize and localize objects in the test images.

The recognition process proposed in this paper is using the following steps.

1. **KPP Extraction**: KPPs are extracted from the test image just like what has been done to the training images, as has been introduced in Sec. 3.1 and Sec. 3.2.
2. **Vote Casting**: a Generalized Hough Transform is performed in searching for possible appearance of the target object. Each KPP found in test image casts votes in a series of continuous 3D voting spaces according to its voting model. Details of this step can be found in Sec. 3.3.2.
3. **Hypothesis Building**: maxima in the voting spaces are found by using Mean-Shift Mode Estimation (MSME) [30] with balloon density estimator, as will be presented in Sec. 4.1. Each maximum found by MSME is corresponding to a round hypothesis containing the possible object position, size and orientation.
4. **Bound Box Generation and Ranking**: bound box hypotheses are generated and ranked according to the round hypotheses by employing class-specific HoG models which are trained with linear SVM. Sec. 4.2 will give details on this step.

Fig.5 demonstrates the process of object recognition with Pair Hough Model.

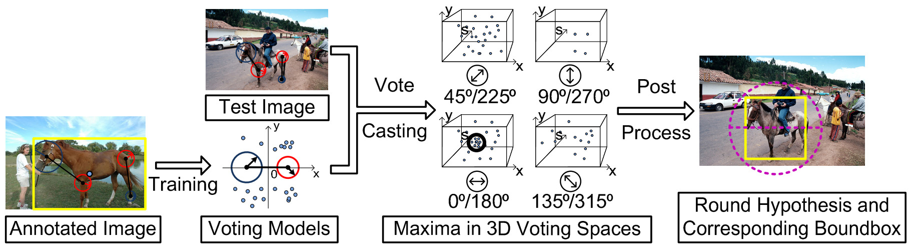


Figure 5: Process of object recognition with Pair Hough Model.

The recognition framework presented in this section can easily work with different interest point detectors and different local descriptors. In the experiments, we tested PHM with a large range of local features, including SIFT [13], SURF [14], ORB [18] and BRISK [17]. Among the features we have been tested, binary descriptors are significantly faster than other local descriptors, but SIFT achieved the best performance.

4.1. Recognition Approach with Mean-Shift Mode Estimation

After votes are casted according to Sec.3.3, the maxima in 3D voting spaces are found by using Mean-Shift Mode Estimation.

4.1.1. Balloon density estimator

Since objects in an image may vary greatly in size, the bandwidth should be chosen carefully in order to both tolerate slight deformation of objects and separate objects which are close to each other. To avoid manually choosing bandwidth for every image, the following *balloon density estimator* [31] is employed, given n votes each denoted by $\mathbf{x}_i = (x_i, y_i, s_i)^T$ and weighting w_i:

$$\hat{f}(\mathbf{x}) = \frac{1}{s^{2q}} \sum_{i=1}^{n} w_i k_i, \quad k_i = k\left(\left\| \tfrac{\mathbf{x}-\mathbf{x}_i}{h_0 s} \right\|^2\right) \tag{11}$$

In equation (11), h_0 and q are two parameters which need to be tuned during training. h_0 is the *error span rate* which is the ratio between bandwidth and the predicted scale, and it takes value in $(0, 1]$. q is the *pair density power* that indicates the relationship between predicted scale and the number of KPPs within the area occupied by the target object. k is the smoothing kernel of a kernel density estimator, and in this paper Epanechnikov kernel [32], which is shown in Eqn.(12), is chosen for efficiency.

$$k(x) = \begin{cases} 1 - x & 0 \leq x \leq 1 \\ 0 & x > 1 \end{cases} \tag{12}$$

Let $g(x) = -k'(x)$, $g_i = g\left(\|(\mathbf{x} - \mathbf{x}_i)/(h_0 s)\|^2\right)$ and $C_q = \frac{2}{h_0^2 s^{2q+2}} \left[\sum_{i=1}^{n} w_i g_i\right]$, then the partial derivatives of $\hat{f}(\mathbf{x})$ become

$$\frac{\hat{\partial} f(\mathbf{x})}{\partial \beta} = C_q \left(\frac{\sum\limits_{i=1}^{n} \beta_i w_i g_i}{\sum\limits_{i=1}^{n} w_i g_i} - \beta \right), \beta \in \{x, y\} \tag{13}$$

$$\frac{\hat{\partial} f(\mathbf{x})}{\partial s} = C_q \left\{ \frac{\sum\limits_{i=1}^{n} w_i \left[g_i \left(\frac{\|\mathbf{x}-\mathbf{x}_i\|^2}{s} + s_i \right) - q h_0^2 s k_i \right]}{\sum\limits_{i=1}^{n} w_i g_i} - s \right\}. \tag{14}$$

Therefore, the expression of mean shift vector becomes

$$\mathbf{m}_{h_0,k} = \begin{pmatrix} \dfrac{\sum_{i=1}^{n} x_i w_i g_i}{\sum_{i=1}^{n} w_i g_i} \\ \dfrac{\sum_{i=1}^{n} y_i w_i g_i}{\sum_{i=1}^{n} w_i g_i} \\ \dfrac{\sum_{i=1}^{n} w_i \left[g_i \left(\frac{\|\mathbf{x}-\mathbf{x}_i\|^2}{s} + s_i \right) - q h_0^2 s k_i \right]}{\sum_{i=1}^{n} w_i g_i} \end{pmatrix} - \mathbf{x}. \tag{15}$$

Intuitively, the balloon density estimator is designed with the idea that error and deformation may cause votes to span a wider area when the size of an object increased. However, without a restriction of s, a balloon density estimator tends to make the predicted size much larger than the real size. Pair density power q works as a penalty that prevents the predicted object area from growing overly large.

4.1.2. Determination of Orientation

The voting spaces of PHM are 3-dimensional, so only coordinates (x, y) and size s can be determined through the MSME optimization, which leaves a the orientation to be determined in another way.

An obvious way of assigning orientation to a maximum is directly using orientation label of the voting space in which this maximum is found. However, orientations assigned this way are too coarse since there are only four possible values for a.

In order to get finer orientation assignment, a_{vote} of all the voting points in a certain neighborhood of the maximum are averaged out and the result is assigned to a of this maximum. For the experiments in Sec.5, the radius of neighborhood is set to half of the bandwidth $(0.5h)$, so it varies along with the bandwidth of balloon density estimator.

4.2. Post Process

After maxima in the voting space have been located, several post processes need to be performed in order to refine the results of MSME and generate bound box hypotheses. First, in order to compare the results of PHM with other state-of-the-art methods, the round hypotheses need to be translated into bound boxes. Second, maxima found by MSME will be ranked before

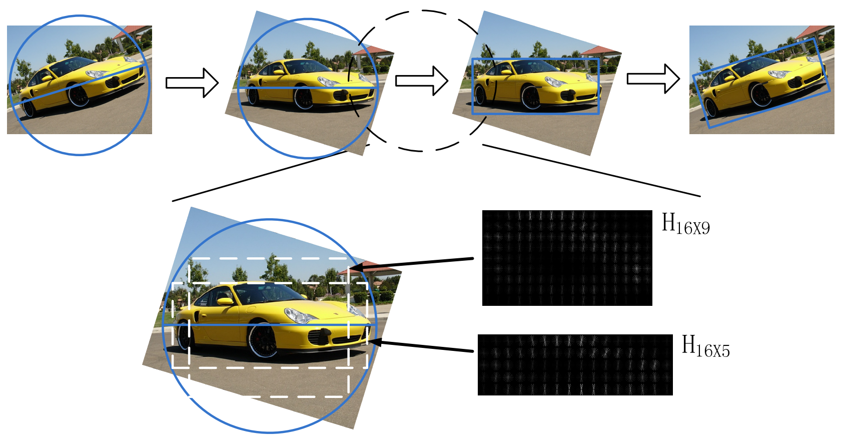


Figure 6: Generation of Bound Box from Round Hypothesis. In the images, circles in solid line show the original round hypothesis. Boxes in dotted line show the possible bound boxes, and boxes in solid line show the result bound boxes.

output, so that popular evaluation measures can be calculated for the results. In the following post process, these two goals are achieved simultaneously by extracting HoG descriptors out of possible bound boxes and compare them with a few trained models.

After a round hypothesis is found by the recognition approach in Sec.4.1, the image is rotated according to the orientation of predicted object. A series of possible bound boxes are then generated within the round hypothesis, in which HoG descriptors are extracted and ranked by a Linear SVM classifier. The bound box with highest rank is selected as the result bound box. After a bound box hypothesis is generated, it is rotated again to fit into the original image.

The grid used to extract HoG descriptor from a bound box is determined by the ratio of width to height of the bound box. Assuming $L \times L$ is the maximum dimension of HoG grids, the dimension $(N \times M)$ of a grid can be calculated with the following equation.

$$N = \begin{cases} L & W > H \\ max\{[\frac{WL}{H}], 1\} & W \leqslant H \end{cases} \tag{16}$$

$$M = \begin{cases} max\{[\frac{HL}{W}], 1\} & W > H \\ L & W \leqslant H \end{cases} \tag{17}$$

In Equation (16) and (17), W and H represent width and height of the corresponding bound box respectively. $[a]$ means the nearest integer of a.

An extracted HoG descriptor is then compared with one of the trained models according to its dimensions, as shown in Fig.6. To train the HoG models for an object category, HoG descriptor is extracted from each ground truth bound box belonging to the target category with a grid depending on the shape of the bound box. Then, linear SVM is used to train model $H_{N\times M}$ for descriptors with a same grid $(N \times M)$.

Post process described in this section requires transformations on the original image, which could add a heavy load to the whole recognition process. However, in practice, real image transformations are not necessary, and instead the HoG calculation could be modified to handle rotated grid. Since the scale of image are not changed during the process, the alternative method will not damage the precision.

5. Evaluation

In order to evaluate the performance of Pair Hough Model and compare it to state-of-the-art methods, several experiments have been conducted and their results will be presented and discussed in this section.

Experiments in this section can be categorized into two sets:

1. **Object Detection Performance** experiments compare the overall precision of all competing methods on a few public data sets and rotated data sets derived from them;
2. **Rotation Adaptivity Analysis** experiments are focused on analyzing the performance of competing methods on different rotated data sets in order to evaluate their rotation adaptive characteristic.

5.1. Data Sets and Measurement

There are many public data sets purposely built for object detection task, and the following challenging data sets are chosen for our experiments.

UIUC Cars. The UIUC cars data set [33] contains side views of cars collected at UIUC, including 1050 training images (550 positive and 500 negative), 170 single-scale test images and 108 multi-scale test images. Images in this data set may contain partially occluded cars and cluttered backgrounds, but the whole data set is still relatively clean and easy.

INRIA Person. The INRIA Person data set [23] has been used for verification of many pedestrian or object detection techniques, including HoG and CHF. This data set contains images of front, back and side views of people, including 1832 training images (614 positive and 1218 negative) and 741 test images (288 positive and 453 negative). INRIA data set also contains training and test images scaled into a same scales (70×134 or 64×128).

PASCAL VOC Challenge. From 2005 to 2012, PASCAL Visual Object Classes [6] Challenges were held every year and provided a series of image sets with high complexity and difficulty for object classification, detection and segmentation tasks. The VOC2007 data set contains 5011 training/validation images and 4952 test images, and unlike its successors the VOC2007 data set contains annotation information of test images, which makes it suitable for our experiments. Images in VOC data sets are complex, cluttered and therefore very close to every day pictures, which makes the object detection task extremely difficult on these data sets. In our experiments, both training and validation data contained in the development kit are used for training and the test data is used for testing just as recommended.

Evaluation measures used in the following sections are the Area of Overlap (AO) and Average Precision (AP).

$$AO = \frac{|A_{gt} \cap A_p|}{|A_{gt} \cup A_p|} \tag{18}$$

$$AP = \frac{\sum\limits_{k=1}^{K} (P(k) \cdot R(k))}{\text{All Ground Truth Objects}} \tag{19}$$

In Equation (18), A_{gt} is the object area given by the ground truth and A_p is the predicted object area. In Equation (19), K is the number of predictions, $P(k)$ is the precision of the first k predictions and $R(k)$ is an indicator which takes value 1 when the kth prediction is correct and 0 otherwise. In PASCAL VOC Challanges and our experiments, a prediction needs to have an AO over 0.5 to be considered correct.

5.2. Pre-process of Data

In order to achieve the best runtime performance of PHM as well as other methods, public data sets need to go through a few pre-processes before the experiments.

5.2.1. Separation of different views

For most categories except symmetric ones like balls, an object viewed from different directions may look distinct, which makes it difficult to train a single Hough model to recognize appearances of an object in different poses. Since all poses cannot be treated the same when recognizing an object, different poses of an object are treated as different objects in the experiments, which means that an object category may be mentioned as "horses facing left" but not merely "horses". In this way, intra-class variances could become low enough for the use of Hough transform based methods. Moreover, four different view tags are used in the following experiments, which are "Frontal", "Rear", "Left" and "Right" in PASCAL definition [6].

5.2.2. Handling the truncated objects

In order to obtain a relatively accurate center and size of an object, the object needs to be completely within the view port of the image. One way to deal with this problem is to train an individual model for each possible truncated object part. We tested this idea on a limited set, and the results are shown in Tab.1.

Table 1: Results of Training Truncated Models.

Number of Models	4	20	24	28	32	36	40
AP (%)	53.5	55.0	55.4	55.2	54.9	53.3	51.0
Run Time(s)	2095	11133	13342	14690	17686	19246	21178

At first, the overall precision increased slowly as the number of models increasing. However, after the models reached a certain number, the precision dropped pretty quickly. Meanwhile, run time increases almost directly proportionally to the number of models. The improvement of performance is relatively less significant than run time increment.

Therefore, when using VOC data sets, incomplete object appearances are discarded during training, i.e. ground truth objects marked as "truncated" are not used. On the other hand, occluded ground truth objects are treated as valid training samples.

5.3. Object Detection Performance

Experiments in this section are designed to compare the overall precision of all competing methods. Data sets used in the following experiments are

UIUC cars, INRIA person, VOC2007 and rotated data sets derived from them. Except for the data sets, the experiments are no different from solving a standard object detection problem.

Because Deformable Part Model [4]), Implicit Shape Model [2], Class-specific Hough Forest [3] and Selective Search [5] are not designed to handle rotated objects, simply comparing them to rotation adaptive methods is not very convincing. Therefore, for each non-rotation-adaptive method, a modified version of the algorithm, which increases rotation adaptivity of the original method, will be tested as well. The detailed modification strategy will be presented in Sec.5.3.2.

In this and the following sections, DPM stands for *Deformable Part Model*, ISM stands for *Implicit Shape Model* , CHF stands for *Class-specific Hough Forest* , SS stands for *Selective Search* , and BRF stands for the method proposed in [29]. PHM is our method Pair Hough Model, and PHMs is an alternative version of Pair Hough Model where voting subjects are individual key points instead of key point pairs.

5.3.1. Generation of rotated data sets

Besides standard object detection task performance, PHM is focused on the adaptivity of rotation. Therefore, for each original data set (UIUC, INRIA or VOC2007), a rotated version is derived from it and used in the following experiments in order to have more rotated objects in test images. In more specific terms, the training set of each original data set is used directly as training set in the corresponding rotated data set, while test images are derived by rotating the original test images by a series of angles (from 0^o to 350^o with a step of 10^o).

During the rotation of images, their corresponding ground truth bound boxes are also transformed. And to be fair, since other methods cannot detect the rotation of objects and will always give results in 4-tuples $(x, y, width, height)$, each ground truth bound box in rotated images is also given by a 4-tuple corresponding to the rectangle covering the transformed bound box and having minimal area, as in Fig.7.

Apparently, automatically generated bound boxes are not so accurate as manual annotation. However, the number of rotated images is extremely large, and manually annotating all of them is approaching impossible for us. Although bound boxes generated in the described way may sometimes be slightly larger than necessary depending on the object shapes, they will always cover the entire objects. Most of the competing methods such as DPM

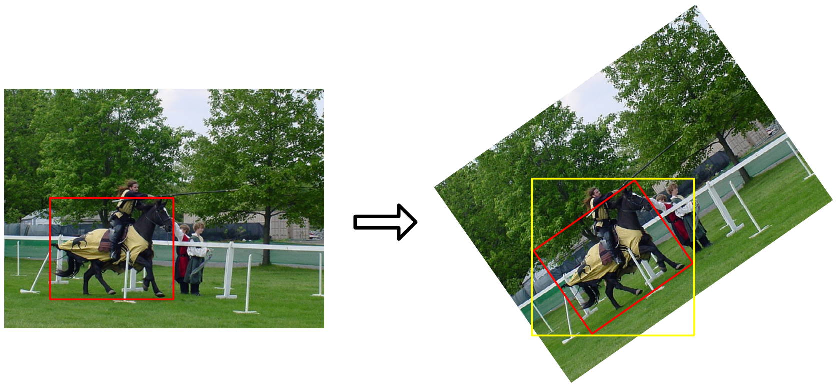


Figure 7: Relationship between ground truth bound boxes in the rotated and original images. Red boxes marks the original bound box and its transformed position in the rotated image. Yellow box marks the bound box actually used in the experiments.

do not require exact ground truth bound boxes as long as the bound boxes cover objects entirely, so the slight inaccuracy of automatically generated bound boxes will not necessarily affect their performance.

5.3.2. Modification of non-rotation-adaptive methods

Since DPM, ISM, CHF and SS have no coping mechanism with object rotation, a modified version of each algorithm will be tested in the following experiments.

The modification mainly consists of three parts:

1. Training an individual model on each rotated training set (from 0^o to 350^o with a step of 10^o),
2. Applying all trained models on each test image during testing and
3. Combining results given by different models into on a final output. As to the combination of results, if two output bound boxes have an AO (described in Sec.5.1) over 0.8, the one with lower score will be discarded in order to remove redundancy from the final output.

Obviously, the modification described here does not change the basic algorithm but combines multiple models into a single functioning framework.

5.3.3. Experimental results and discussion

All competing methods (unmodified and modified versions) are applied to the rotated test sets as well as the original ones so that their standard object detection performance and tolerance of transformations can be evaluated.

Table 2: Object Detection Results in AP (%) from Unmodified Methods.

Data Set	**DPM**	**ISM**	**CHF**	**SS**	**BRF**	**PHM**	**PHMs**
INRIA	**91.1**	86.0	80.7	87.2	78.9	86.5	80.2
UIUC Single	94.2	90.0	**96.3**	96.1	94.3	95.6	87.3
UIUC Multi	93.9	87.6	**96.0**	94.8	93.5	95.1	83.0
INRIA (r)	10.9	8.5	8.3	8.7	67.1	**69.4**	59.7
UIUC Single (r)	11.3	8.7	10.2	10.7	**76.8**	76.0	66.8
UIUC Multi (r)	11.8	8.5	9.6	10.1	70.5	**72.3**	65.5

Table 3: Object Detection Results in AP (%) from Modified Methods.

Data Set	**DPM**	**ISM**	**CHF**	**SS**	**BRF**	**PHM**	**PHMs**
INRIA (r)	62.7	57.8	52.7	57.8	67.1	**69.4**	59.7
UIUC Single (r)	62.8	58.8	65.1	64.3	**76.8**	76.0	66.8
UIUC Multi (r)	64.1	59.6	65.7	65.2	70.5	**72.3**	65.5

Although PHM has not achieved the best (but comparable) results when applied to the original data sets, which rarely contain rotated objects, it outperforms state-of-the-art methods on the rotated data sets, as shown in Tab.2 and Fig.8(a)(b). Also, PHM (key point pairs) significantly outperforms PHMs (individual key points) on almost all the data sets, which proves the importance of pair based voting.

Fig.8(c) illustrates that by modifying the original algorithms into rotation adaptive version, the performance of DPM, ISM, CHF and SS can be significantly improved on rotated data sets. Nevertheless, while using multiple models increases overall precision on rotated data sets, it also introduces more random errors per image, which leads to the phenomenon that the APs of a modified method are usually slightly lower than the APs of its counterpart on the original data sets. Therefore, on the rotated data sets, methods that are designed to handle rotated objects (BRF and PHM) still outperform the modified methods.

Meanwhile, online (test) run time increases as the number of models increasing, which means some of the modified methods are even slower than PHM, which is the slowest among the original methods, e.g. the time needed by the modified Selective Search to process the entire rotated VOC2007 data set is almost 7 times of that needed by PHM.

Fig.9 shows a few samples of rotated objects in PASCAL VOC2007 data

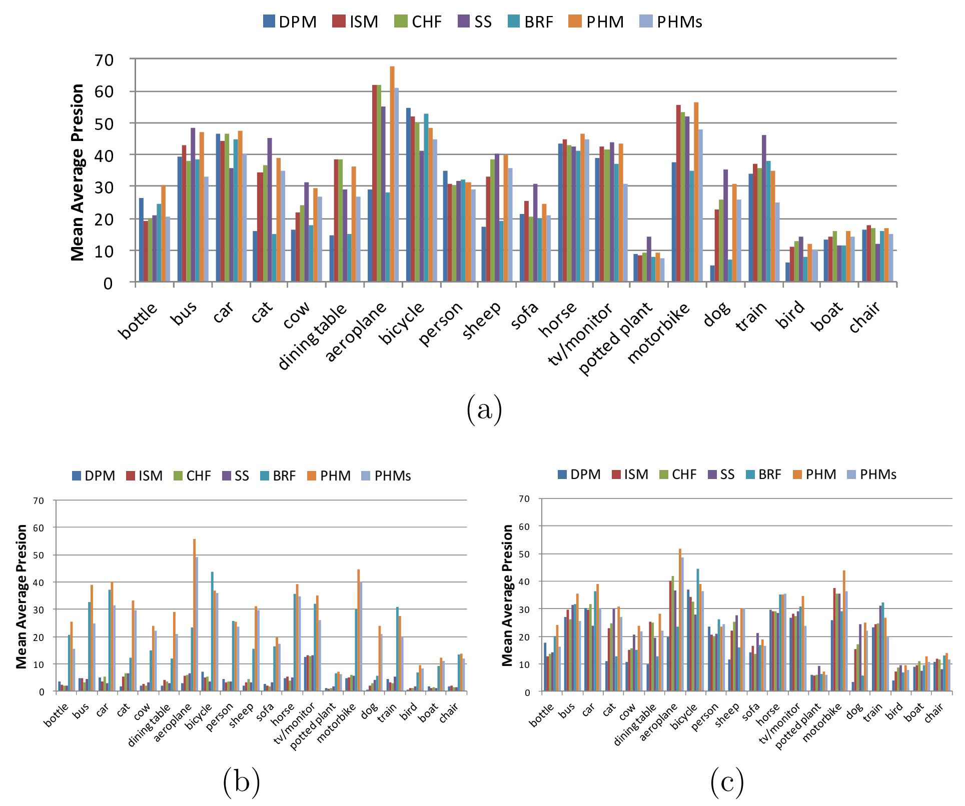


Figure 8: Object Detection Results on PASCAL VOC2007 Image Set in AP (%). (a) gives the results on original images, while (b) and (c) show the results on rotated images. Moreover, statistics in (b) and (c) are yielded by unmodified methods and modified methods respectively.

set and their corresponding recognition results of different methods.

5.4. Rotation Adaptivity Analysis

For more detailed analysis on the rotation adaptivity of the competing methods, we designed the following experiments to show the changing of performance while images and objects rotate. The rotated data set derived from VOC2007 data set is used here. Figures in this section can be seen as expansions of Fig.8(b)(c).

As shown in Fig.10, DPM, ISM, CHF and SS can only maintain their accuracy in a relatively small range of angle. It is easy to explain:

- ISM and CHF are based on monochrome image patches, which are obviously very sensitive to rotation of images. The binary tests used by

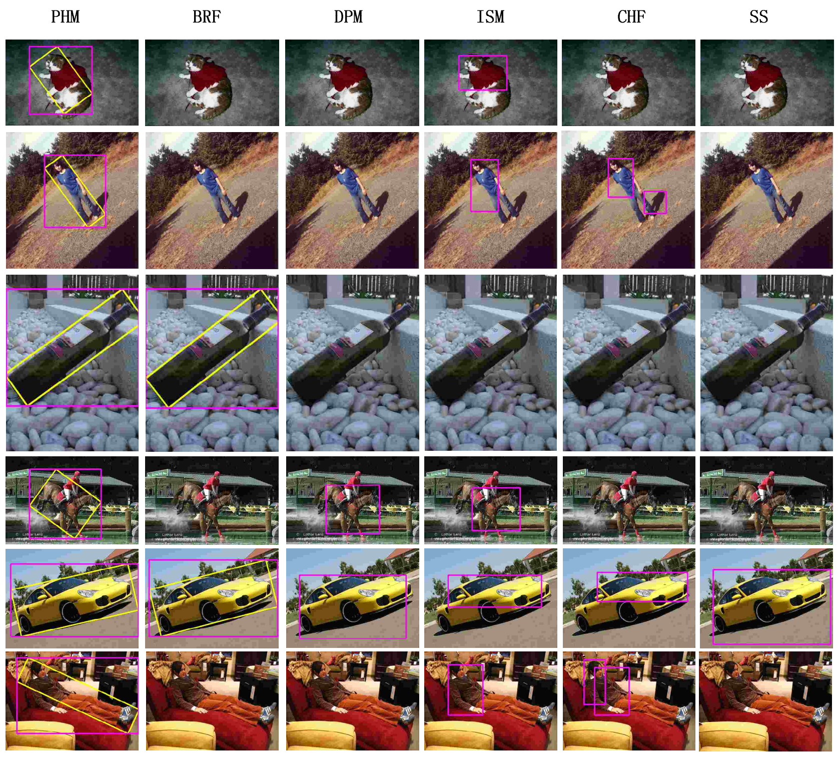
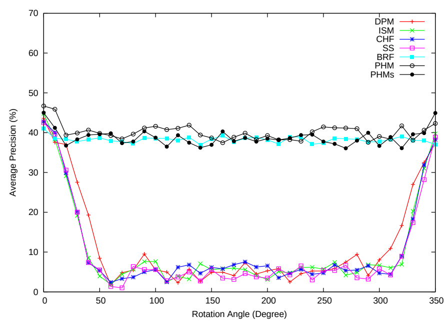


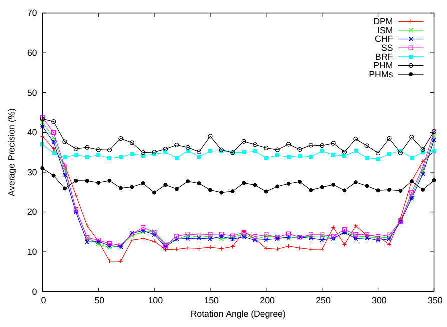
Figure 9: Recognition results of rotated objects in PASCAL VOC2007 data set.

CHF are comparisons between two pixels in a patch and ISM uses direct patch-wise comparisons, so CHF is relatively more rotation tolerant than ISM.

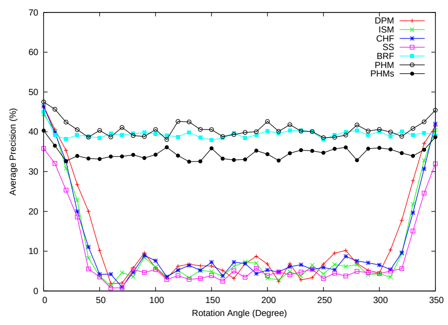
- DPM is based on HoG descriptor, whose rotation tolerance depends on the orientation discretization, i.e. the width of bins in the orientation histogram of HoG descriptor. Latent-SVM [4] can provide tolerance of rotation to some degree if trained with slightly rotated objects, but this method works only in a limited range. Soft assignment can improve its overall performance but has little influence on the rotation adaptivity.
- The segmentation and area merging step of the Selective Search algorithm is rotation invariant, but its ranking step involves a 4-leveled spatial pyramid matching, which makes it highly sensitive to any kind of deformation. Training with rotated images can improve the rotation tolerance of Selective Search, but it also renders the upper levels



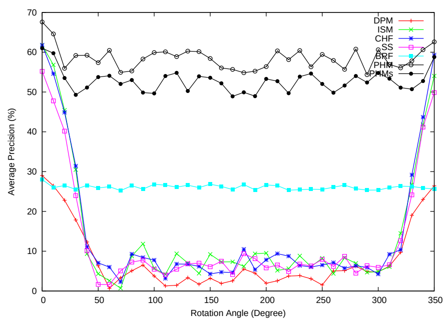
(a) horse



(b) tv/monitor



(c) car



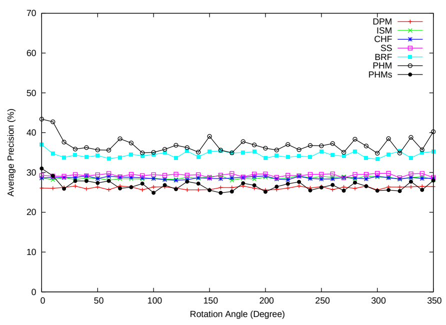
(d) aeroplane

Figure 10: Demonstration of Rotation Adaptivity Analysis results of unmodified methods while the images are rotated clockwise.

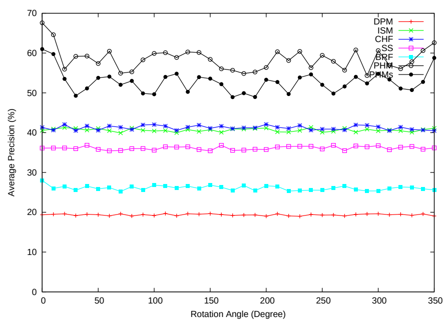
of the spatial pyramid nearly useless during classification and degenerates the spatial pyramid into bag-of-words, which damages the overall performance greatly.

As shown in Fig.11, the modified DPM, ISM, CHF and SS seem even more stable than BRF and PHM while images rotating, but the stability comes with the price of tedious work of training models on rotated data sets and longer run time in both training and testing.

By contrast, even without training on rotated images BRF and PHM perform almost the same while the test images rotating. PHM is slightly less stable than BRF due to the instability of interest point detectors. However, PHM achieved much better overall results on the original data sets, so on the rotated data sets PHM also outperforms BRF.



(a) tv/monitor



(b) aeroplane

Figure 11: Demonstration of Rotation Adaptivity Analysis results of modified methods while the images are rotated clockwise.

5.5. Efficiency Issue

Using KPPs instead of individual key points leads to an efficiency problem because if every two key points form a pair then there will be numerous pairs per image. Too many KPPs would cast too many votes and eventually slow down the MSME solution. In this section, the efficiency issue of PHM will be discussed thoroughly.

5.5.1. Relationship between the number of key points and the number of votes

The number of casted votes per image is one of the most important factors that affect the efficiency of PHM. Since every two key points can form a pair, the number of votes can be expected to be proportional to N^2, where N is the number of key points in an image. However, in practice, the 6-tuple which is used to represent and compare KPPs holds a fairly tight restriction, so there are hardly too many votes being casted per image.

Table 4: Average numbers of key points and casted votes per image.

Data Set / Category	**DoG Detector**		**Fast Hessian**	
	Key Points	**Votes**	**Key Points**	**Votes**
VOC2007 / horse	485	9351	285	5231
VOC2007 / car	464	5328	264	3758
VOC2007 / tv/monitor	398	3197	198	2902
INRIA Person	253	4377	153	3170
UIUC Cars	72	336	32	359

Tab.4 shows the average number of key points and votes per image in several data sets according to our experiments. The actual number of votes is much smaller than the expected number.

5.5.2. Grid Seeding

To detect multiple objects within one image, multiple seed points are used during the MSME optimization process. Random seeding is simple but a large number of seed points need to be chosen in order to achieve high accuracy. Therefore, in our experiments Grid Seeding is used for this task.

First, a discrete Hough voting is done in a 3D grid, which has 3 dimensions representing x, y and s respectively. Then cells with local maximum values are found in the grid and their center points are used as seed points.

By using Grid Seeding, the number of seed points can be limited while not damaging the accuracy. And also, since the seed points are fairly close to the actual maxima in the continuous voting space, the iteration process of MSME will finish very quickly.

5.5.3. Comparison among Hough based methods

The following table gives the average run times per image of competing methods. All the tests ran on a DELL Precision T7800 with two Intel(R) Xeon(R) E5-2609 2.40GHz CPUs, and all the tests ran in single thread.

Table 5: Average run times per image of different Hough based methods in seconds.

Data Set / Category	**ISM**	**CHF**	**PHM with SIFT**			
			Total	**Vote**	**MSME**	**Post**
VOC2007 / bottle	0.52	0.60	1.24	0.82	0.12	0.3
VOC2007 / aeroplane	0.55	0.59	1.13	0.74	0.09	0.3
VOC2007 / boat	0.48	0.56	1.18	0.70	0.08	0.4
INRIA Person	0.52	0.61	1.14	0.71	0.13	0.3
UIUC Cars	0.13	0.15	0.24	0.13	0.01	0.1

As shown in Table 5, compared to other Hough based methods, PHM is indeed a little time consuming but not insufferable. Also, the most time consuming part of PHM is the vote casting process which can easily be parallelized or speed up with other optimization techniques. In our experiments, **Vote** part achieved nearly linear speed up on Thread Building Blocks (TBB) with respect to the number of processor cores.

6. Conclusion

This paper presented a novel scaling and rotation adaptive object detection method and evaluated its performance through experiments on a series of data sets. The experiment results prove the favorable accuracy and transformation adaptivity of the proposed method. On the whole, PHM is capable of achieving high performance on a range of object categories and is highly robust to rotation of objects.

Our future work is to address a few problems to improve PHM, including the following subjects:

- *Part-based Reasoning*: since PHM is still a little naive and has no reasoning mechanism, when used for recognizing complicated objects with conjoint parts such as people, dogs and cats, PHM usually has a very high false positive ratio. To reduce the false positive ratio when recognizing complex objects, a part-based reasoning mechanism is needed.
- *Accurate Orientation Prediction*: accurate orientation prediction with bound box annotation is a challenging task, and once it is solved a 4D voting space could be used to replace the current 3D voting space, which could improve the performance of PHM.
- *Combination with Active Learning Technique*: experiments described in this paper did not make use of the incomplete objects in the training sets, which significantly limits the actual size of the training set. By combining with Active Learning Techniques [34], discarded training samples may become useful and increase the performance of PHM.

7. Acknowledgement

This work is partially supported by the National Natural Science Foundation of China (Grant Nos: 61402026 and 61370125).

Reference

[1] D. Ballard, Generalizing the hough transform to detect arbitrary shapes, Pattern Recognition 13 (2) (1981) 111 – 122.

[2] B. Leibe, A. Leonardis, B. Schiele, Robust object detection with interleaved categorization and segmentation, International Journal of Computer Vision 77 (1-3) (2008) 259–289.

[3] J. Gall, V. Lempitsky, Class-specific hough forests for object detection, in: In Proceedings IEEE Conference Computer Vision and Pattern Recognition, 2009.

[4] P. Felzenszwalb, D. McAllester, D. Ramanan, A discriminatively trained, multiscale, deformable part model, in: Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on, 2008, pp. 1–8.

[5] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, A. W. M. Smeulders, Selective search for object recognition, International Journal of Computer Vision 104 (2) (2013) 154–171.

[6] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, A. Zisserman, The pascal visual object classes (voc) challenge, International Journal of Computer Vision 88 (2) (2010) 303–338.

[7] M. Pandey, S. Lazebnik, Scene recognition and weakly supervised object localization with deformable part-based models, in: Computer Vision (ICCV), 2011 IEEE International Conference on, 2011, pp. 1307–1314.

[8] X. Liu, Y. Lou, A. W. Yu, B. Lang, Search by mobile image based on visual and spatial consistency, in: Multimedia and Expo (ICME), 2011 IEEE International Conference on, 2011, pp. 1–6.

[9] J. He, J. Feng, X. Liu, T. Cheng, T.-H. Lin, H. Chung, S.-F. Chang, Mobile product search with bag of hash bits and boundary reranking, in: Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, 2012, pp. 3005–3012.

[10] X. Liu, B. Lang, Y. Xu, B. Cheng, Feature grouping and local soft match for mobile visual search, Pattern Recognition Letters 33 (3) (2012) 239 – 246.

[11] B. Leibe, B. Schiele, Scale-invariant object categorization using a scale-adaptive mean-shift search, in: Pattern Recognition, Vol. 3175 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2004, pp. 145–153.

[12] T. Lindeberg, Feature detection with automatic scale selection, International Journal of Computer Vision 30 (1998) 79–116.

[13] D. Lowe, Object recognition from local scale-invariant features, in: Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on, Vol. 2, 1999, pp. 1150–1157 vol.2.

[14] H. Bay, T. Tuytelaars, L. V. Gool, Surf: Speeded up robust features, in: In ECCV, 2006, pp. 404–417.

[15] K. Mikolajczyk, C. Schmid, An affine invariant interest point detector, in: A. Heyden, G. Sparr, M. Nielsen, P. Johansen (Eds.), Computer Vision, ECCV 2002, Vol. 2350 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2002, pp. 128–142.

[16] T. Tuytelaars, L. V. Gool, Wide baseline stereo matching based on local, affinely invariant regions, in: In Proc. BMVC, 2000, pp. 412–425.

[17] S. Leutenegger, M. Chli, R. Siegwart, Brisk: Binary robust invariant scalable keypoints, in: Computer Vision (ICCV), 2011 IEEE International Conference on, 2011, pp. 2548–2555.

[18] E. Rublee, V. Rabaud, K. Konolige, G. Bradski, Orb: An efficient alternative to sift or surf, in: Computer Vision (ICCV), 2011 IEEE International Conference on, 2011, pp. 2564–2571.

[19] A. Alahi, R. Ortiz, P. Vandergheynst, Freak: Fast retina keypoint, in: Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, 2012, pp. 510–517.

[20] D. Maio, D. Maltoni, Real-time face location on gray-scale static images, Pattern Recognition 33 (9) (2000) 1525 – 1539.

[21] A. Samal, P. A. Iyengar, Human face detection using silhouttes, International Journal of Pattern Recognition and Artificial Intelligence 09 (06) (1995) 845–867.

[22] G. Chow, X. Li, Towards a system for automatic facial feature detection, Pattern Recognition 26 (12) (1993) 1739 – 1755.

[23] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, Vol. 1, 2005, pp. 886–893.

[24] C. Schmid, Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories, in: In CVPR, 2006.

[25] V. Kyrki, J.-K. Kamarainen, H. Kälviäinen, Simple gabor feature space for invariant object recognition, Pattern recognition letters 25 (3) (2004) 311–318.

[26] K. Liu, H. Skibbe, T. Schmidt, T. Blein, K. Palme, T. Brox, O. Ronneberger, Rotation-invariant hog descriptors using fourier analysis in polar and spherical coordinates, International Journal of Computer Vision 106 (3) (2014) 342–364.

[27] J. Ruiz-Pinales, J. J. Acosta-Reyes, R. Jaime-Rivas, A. Salazar-Garibay, Rotation invariant image recognition using hough transform and support vector machines, in: Electronics and Photonics, 2006. MEP 2006. Multiconference on, IEEE, 2006, pp. 196–198.

[28] X. Yang, C. Liao, Q. Liu, K.-T. Cheng, Minimum correspondence sets for improving large-scale augmented paper., in: VRCAI, 2011, pp. 59–68.

[29] M. Villamizar, F. Moreno-Noguer, J. Andrade-Cetto, A. Sanfeliu, Efficient rotation invariant object detection using boosted random ferns, in: Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, IEEE, 2010, pp. 1038–1045.

[30] D. Comaniciu, P. Meer, Mean shift: a robust approach toward feature space analysis, Pattern Analysis and Machine Intelligence, IEEE Transactions on 24 (5) (2002) 603–619.

[31] D. Comaniciu, V. Ramesh, P. Meer, The variable bandwidth mean shift and data-driven scale selection, in: Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on, Vol. 1, 2001, pp. 438–445 vol.1.

[32] D. W. Scott, Multivariate density estimation: theory, practice, and visualization, Vol. 383, Wiley. com, 2009.

[33] S. Agarwal, A. Awan, D. Roth, I. C. Society, Learning to detect objects in images via a sparse, part-based representation, IEEE Transactions on Pattern Analysis and Machine Intelligence 26 (2004) 2004.

[34] L. Huang, X. Liu, B. Ma, B. Lang, Online semi-supervised annotation via a proxy-based local consistency propagation, Neurocomputing 149, Part C (2015) 1573 – 1586.

Appendix: Proofs of proposition 3.1 and 3.2

Proposition 3.1: When an image is converted into another image by multiplying the coordinates of each of its pixel by a linear transformation matrix $\mathbf{A}$ in the form of Eqn(5), if the interest point detector and descriptor are scale and rotation invariant, a KPP (kp_1, kp_2) in the original image and its corresponding KPP (kp'_1, kp'_2) in the transformed image will be assigned the same 6-tuple after the normalization.

$$\mathbf{A} = \mathbf{TSR} = \begin{pmatrix} 1 & 0 & \Delta_1 \\ 0 & 1 & \Delta_2 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \gamma & 0 & 0 \\ 0 & \gamma & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\delta & -\sin\delta & 0 \\ \sin\delta & \cos\delta & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{20}$$

Proof: Let $\mathbf{x}/\mathbf{x}'$ be the position of kp/kp', s/s' be the scale of kp/kp', and η/η' be the angle between x axis and the vector from the origin to kp/kp', the following equations can be easily derived from the definition of linear transformation.

$$\mathbf{x}' = \mathbf{A}^{\mathrm{T}}\mathbf{x} \tag{21}$$
$$s' = \gamma \cdot s \tag{22}$$
$$\eta' = \eta + \delta \tag{23}$$

According to Fig.4 and Eqn(21)-(23), if (kp_1, kp_2) is a KPP in the original image and (kp'_1, kp'_2) is its corresponding one in the transformed image, then

$$\mathbf{x}'_C = \mathbf{A}\mathbf{x}_C = \begin{pmatrix} \gamma \cdot [(x_C + \Delta_1) \cdot \cos\delta - (y_C + \Delta_2) \cdot \sin\delta] \\ \gamma \cdot [(x_C + \Delta_1) \cdot \sin\delta + (y_C + \Delta_2) \cdot \cos\delta] \\ 1 \end{pmatrix} \tag{24}$$
$$\varphi' = \varphi + \delta \tag{25}$$
$$R' = \gamma \cdot R \tag{26}$$
$$s'_i = \gamma \cdot s_i, i = 1, 2, \tag{27}$$

so

$$\begin{aligned} a'_i &= \eta'_i - \varphi' \\ &= \eta_i + \delta - \varphi - \delta \\ &= a_i, i = 1,2 \qquad (28) \\ s'_{mi} &= s'_i / R' \\ &= \frac{\gamma \cdot s_i}{\gamma \cdot R} \\ &= s_{mi}, i = 1,2. \qquad (29) \end{aligned}$$

In the above equations, $\mathbf{x}_C = (x_C, y_C, 1)^T$ and $\mathbf{x}'_C$ are the positions of the center point C in the original image and the transformed image, and a_1/a_2, s_1/s_1, s_{m1}/s_{m2}, φ and R are the same as defined in P_M and Fig.4.

And, when the descriptor is scale invariant, c_i will be the same after any kind of rigid-body transformation. Therefore, (kp_1, kp_2) and (kp'_1, kp'_2) will be assigned the same 6-tuple after the normalization. □

Proposition 3.2: When an image is converted into another image by applying the transformation described in Prop.3.1 and the key point detector and descriptor are scale and rotation invariant, if voting models are trained in the original image, during the testing on the transformed image the number of votes casted at the transformed object position, scale and orientation will be at least the same as the number of KPPs in the original image.

Proof: Since for each KPP there will be a vote point in its model space corresponding to the target object, we denote the vote point of P_I by $(r_m, \theta, s_m, a_v, w)$.

After a rigid-body transformation is applied on an image, an object sample in the original image will become

$$\begin{aligned} \mathbf{x}'_o &= \mathbf{A}\mathbf{x}_o = \begin{pmatrix} \gamma \cdot [(x_o + \Delta_1) \cdot \cos\delta - (y_o + \Delta_2) \cdot \sin\delta] \\ \gamma \cdot [(x_o + \Delta_1) \cdot \sin\delta + (y_o + \Delta_2) \cdot \cos\delta] \\ 1 \end{pmatrix} \qquad (30) \\ s' &= \gamma \cdot s \qquad (31) \\ \mu' &= \mu + \delta \qquad (32) \end{aligned}$$

In the above three equations, $\mathbf{x}_o = (x_o, y_o, 1)^T$ and $\mathbf{x}'_o$ are the positions of the object in the original image and the transformed image, and μ and μ'

are the orientations of the object in the original image and the transformed image, i.e. for each KPP there is $\mu = a_v + \varphi$.

When the key point detector and descriptor are scale and rotation invariant, a KPP P_I found in the original image will have a corresponding KPP P'_I in the transformed image, and as proved in Prop.3.1 P_I and P'_I will be mapped to the same P_M. Therefore, during testing P'_I will cast at least one vote according to the vote point corresponding to P_I.

Let $\vartheta(\mathbf{x})$ be the angle between vector $\mathbf{x}$ and the x axis, we get $\vartheta(\mathbf{x}_o - \mathbf{x}_C) = \theta + \varphi$. In addition, according to the definition of r in Fig.4, $r = \|\mathbf{x}_o - \mathbf{x}_C\|$. Therefore,

$$r \cdot \cos(\theta + \varphi) = x_o - x_C \tag{33}$$

$$r \cdot \sin(\theta + \varphi) = y_o - y_C. \tag{34}$$

Therefore, when P'_I casts vote according to $(r_m, \theta, s_m, a_v, w)$, the vote will be casted at

$$\begin{aligned}
x_{vote} &= x'_C + r_m \cdot R' \cdot \cos(\theta + \varphi') \\
&= \gamma \cdot (x_C + \Delta_1) \cdot \cos\delta - \gamma \cdot (y_C + \Delta_2) \cdot \sin\delta + \gamma \cdot r_m \cdot R \cdot \cos(\theta + \varphi + \delta) \\
&= \gamma \cdot [(x_C + \Delta_1) \cdot \cos\delta - (y_C + \Delta_2) \cdot \sin\delta + \\
&\quad r \cdot \cos(\theta + \varphi) \cdot \cos\delta - r \cdot \sin(\theta + \varphi) \cdot \sin\delta] \\
&= \gamma \cdot [(x_C + x_o - x_C + \Delta_1) \cdot \cos\delta - (y_C + y_o - y_C + \Delta_2) \cdot \sin\delta] \\
&= x'_o && (35) \\
y_{vote} &= y'_o && (36) \\
s_{vote} &= s_m \cdot R' \\
&= s_m \cdot R \cdot \gamma \\
&= s \cdot \gamma = s' && (37) \\
a_{vote} &= a_v + \varphi' \\
&= a_v + \varphi + \delta \\
&= \mu + \delta = \mu' && (38)
\end{aligned}$$

The above four equations hold for every P_I in the original image, so there are at least n votes will be casted at the transformed object position, scale and orientation, where n is the number of KPPs in the original image.

$\square$