# OBJECT RETRIEVING FROM IMAGE DATABASE BY AUTOMATIC FEATURE SELECTION

by

NGUYEN THIEN BAO

A THESIS

submitted in partial fulfillment of the requirements for the degree

MASTER OF SCIENCE

Department of Human and Computer Intelligence
College of Information Science and Engineering

RITSUMEIKAN UNIVERSITY
Kyoto, Japan

2011

Approved by:
Major Professor
Yoshiaki Shirai

# Copyright

# Abstract

Nowadays, there are more and more digital medias (digital image, digital video, e-book, etc.). This growing raises an issue in technology: how to retrieve/search images in multimedia databases. A very large number of research works has focused on searching mechanisms in image databases for efficient retrieval. For recent decades, Content-based image retrieval (CBIR) has been regard as the traditional and dominant technique about this problem. The most challenge in one CBIR is semantic gap (the gap between the cognitive concept of an image for human and the information extracted from low level image features). The semantic gap was addressed in many researches. As an attempt to bridge the semantic gap, in stead of based on high-level semantic of an image, CBIR should know which object that image contains. Due to this, in recent years, object detection has been gaining more and more attentions.

This research focuses on developing a system that can retrieval objects from large image database by exploring types of image feature necessary for recognition of common objects in scene. First, we go to proposed methodology for global representation for these features that can be used in learning. Then, we figure out a novel method for generic object detecting in still images. Our method is simple, computationally efficient and bases on features that is easily seen by naked-eye and very close with natural detecting by human. We present experimental results for detecting many visual categories including side view car, front view car, bike, motorbike, train, aero plane, horse, sheep, tower and flower. Results clearly demonstrate that the proposed method is robust and produces good detection accuracy rate.

Moreover, manual identification features is at best time consuming and usually optimistic. Because of that, firstly, feature identification need to be accurate, repeatable and quantitative. Secondly, it is difficult to figure out which feature is good for a type of object. Due to that, the main advantage of this research is that our system can automatically choose features which are best for detecting one type of object. Experiment results shows that average precision and recall between automatically choosing and manually choosing is similar. Until this time, this is the newest approach without manually defining features before detecting stage.

# Table of Contents

# List of Figures

# List of Tables

*To my Suppreme Master Ching Hai,*
*To the spirit of my beloved father,*
*To my mothers,*
*To my love,*

## Acknowledgements

Firstly, I wish to express my sincere appreciation to Professor Yoshiaki Shirai for his supervision. His wise academic advice and ideas have played an extremely important role in the work presented in this thesis. Without Prof. Shirai's support, this thesis would not have been possible. I also would like to be indebted the help of Professor Hiromi Tanaka and Professor Nobutaka Shimada.

Secondly, I would like to thank my friends and colleagues in Computer Vision lab - in particular, Dr. Tadashi Matsuo and Mr. Hiroshi Yamada - for their inspirations and discussions. I would also like to thank two secretaries, Mrs. Mayumi Itou and Mrs. Masami Masuda for their help in my research.

I am also thankful to the staff in the School of Information Science and Engineering, JICE office and two my roommates for their assistance during the past two years I stay in Japan.

Finally, I would like to thank my family for their everlasting love and support. I want to give this as a present for my father who always looks after my steps in life.

# Chapter 1 - Introduction

In the last decade, digital imagery has grown at a phenomenal speed in many directions, resulting in an explosion in the number and size of image archives required to be organized. In particular, with the widespread use of digital cameras, mobile phones with built-in cameras and storage of personal computers reaching to a level of hundreds of gigabytes, individuals nowadays can easily produce thousands of personal images. Meanwhile, photo sharing through the Internet is becoming more and more popular. For example, by June 2005, the Internet photo-sharing website Flickr1 had almost one million registered users and hosted 19.5 million photos, with a growth of about 30 percent per month (Li and Wang, 2006). By April 2007, Flick had over 5 million registered users and over 250 million images (Ames and Naaman, 2007). Although people like the Filickr users are increasingly attaching tags to their images, the vast majority of the images on the Internet are barely documented, making it very difficult for people to find one of interest. In order to handle overload and exploit the massive image information, we need to develop techniques to document and search images. Earlier efforts in image search has been mainly focused on content-based image retrieval (CBIR), which retrieves images by analyzing and comparing low-level image information. CBIR systems usually rely on the feeding of desired image examples from the user as a starting point, and are known as the query by example paradigm. However, unskilled users may find it tedious to do so. Query by semantics (e.g. textual words) is more preferred. For example, if you want to find sunset images, it is more convenient and explicit to directly use the semantic word “sunset” as the query, than to find a sunset image or draw a sunset-like figure first. Intuitively, image retrieval would be more straightforward if all the images in the database were semantically annotated. By using standard text query techniques, images could be found in a manner that would meet the different needs of many users. Moreover, by combining these text-based approaches with visual content search techniques, users could have much more control over the search. In this approach, image semantics are learned from an external source such as user's feedback and textual annotations, and of course both are provided by human beings. In particular, textual annotations are made either for explicit image categorization tasks or implicit writing processes in news, papers, books or web pages. Learning from external sources is limited because we don’t know the annotation is correct or not. Moreover, there are different scenarios in which descriptive annotations are not

available for all images in the collection or the user is not able to provide an accurate textual query. Furthermore, several works have shown that, even in the presence of text descriptions, visual features are a very useful information source to identify relevant images in a large scale CBIR system, showing important improvements in performance [8,10].

But the problem is that semantic of an image is not easy to get until user actually see that image. This is referred as “semantic gap”, the gap between the information that can be extracted from images and the interpretation of the images for humans. As an attempt to bridge the semantic gap, in stead of based on high-level semantic of an image, CBIR should knows which object that image contains. The more object a system can detect, the more precise that system is. Due to this, in recent years, object detection has been gaining more and more attentions. Object detection is a process of identifying and locating objects in a scene. It is a process of combining digital image processing and computer vision (Gonzalez *et al*, 2001). The main goal of an object detection process is to detect and identify objects in a scene. Although different solutions for this matter have been developed over the past few years, but until now object detecting and recognition is still a difficult problem for computer vision. Most of these approaches based on features of an image including both local and global one for detecting object, which were descried in chapter 2. The goal of this research is to develop a necessary methodology for recognition of general object in still images by automatically choosing features. There are many approaches based on both local and global features, but almost these features are difficult to see with human eyes. In our method, we base on features that are easily seen by naked-eye and very close with natural detecting by human. One more thing is that, most state-of-art methods focus on how to recognize object after fixing some features. In this research, we want to let system choose features without handling from a list of features, and the system determines by itself which features are good for detecting a given object.

## 1.1. Aims and objectives

The original aims of this work were to investigate promising approaches to automatic object retreiving from large image database, especially content-based image retrieval, by either utilising and modifying different techniques in the object detection community, or developing new techniques. The efforts toward achieving this objective consist of two intertwined parts: the development of a suitable approach for object detecting based on multi features (including local

and global features), and the development of an advanced technique for automatically choosing good features for every kind of object. In this work, we will explore both areas, though the second is the focus of the research.

In the process of pursuing the objective, the work has also investigated some relevant issues and tasks regarding object detection from static image. For example, we have closely examined some quality issues of the data-sets used for experiments on object regconization. Beside, we have also made a step forward to attempt to collect and manage large image database of serveral kind of objects.

More specifically the aim of the thesis was to:

- Develop a robust object detection approach that are applicable to a wide range of image data.
- Propose and develop a general algorithm for automatically choosing good features for every categorie of object.

## 1.2. Contributions

This thesis brings a number of contributions to the field of automatic image annotation. They are itemised briefly as follows.

- An in depth investigation into some of the quality issues with image data-sets that are used for research on object detecting from static image.
- The demonstration of the potentials of many feature-extract-algorithms such as Haris Coner detector, Hough Transform, SURF-64/128, ...
- The development of a model for descripting image throuhg feature vector space that improves runing-time of object detection.
- The development of an robust approach to finding object classes in an unlabelled still image collection using both local and global features such as edge, corner, HoG, circle, line, color, SURF.
- The development of a newest algorithm (named best-fixed algorithm) for un-manually selecting best features to detect a specific object.
- The implement of the proposed best-fixed algorithm on a large image database with many object categories from natural objects (flower, animal, ...) to man-made ones (car, bike, tower, ...)

- The evaluation of proposed best-fixed algorithm's reslut, comparation with outcome of manually choosing feature.

The research has led to one refereed conference papers and one refereed workshop paper. Bao and Shirai (ISS, Sendai, 2009) demonstrated some problems of experimentations on how to extract features and combinate these features for detecting many kind of objects using the popular image database such as CALTECH, UIUC and PASCAL contest set. Bao and Shirai (PRMU, BKC, 2011) proposed the novel best-fixed algorithm, and gave more detailed experiments and discussions on automatically choosing good features for a kind of object.

## 1.3. Thesis structure

This thesis presents the work carried out by the author in attempting to achieve the goals outlined earlier in Section 1.1. The structure and content of the thesis is described in the following by chapter basis.

- **Chapter 2 - Background** Introduction the background behind this work. Research on traditional content-based image retrieval is introduced, in the form of three different retrieval scenarios. It then goes on to review a number of very different object detection techniques in the literature. Finally it describes several performance evaluation metrics used in this work.
- **Chapter 3 – Feature Extraction** Presents a variety of techniques regarding the description of the information encapsulated in images through features. Every features has different teachniques to extract. The chapter describles detail in technical about how to detect and extract features used in this research such as edge, corner, HoG, Cirle/Line, color, SURF.
- **Chapter 4 – Object Detection Based on Multi Features** Issues with a robust approach for detecting object. In this approach, combination both local and global features is used to get higher accuracy. The object classifier (object/non-object) with every features is also presented detaily in this chapter. The last part of this chapter presents result of proposed method for detecting many kind of objects from natuaral object (horse, sheep) to man made object (car, bike, motorbike, aeroplan, train).
- **Chapter 5 – Automatic Feature Selection** Proposes a novel auto-choosing features technique that incorporates the salient features of every type of objects with a

probabilitical and statistical model. This is the latest algorithm compared with other state-of-the-art detecting approaches that almost pre-define list of features before detecting. Results of object detection based on auto feature selection is also describled. This part ends with two comparasions: result between auto- and manual choose feature; and our system with PASCAL 2010 contest.

- **Chapter 6 - Conclusions and Future Work** Discusses and concludes the overall results and contributions from the work presented in previous chapters. The chapter ends with some pointers to the future work regarding chapter 4 to 6, and an overview of the future of object detection in still image from the author's point of view.

# Chapter 2 - Background

The aim of this chapter is to present an overview of the research that is related to this thesis, with focus on object detection by automatic feature selection technique. Firstly, we briefly review the field of content based image retrieval, which serves as the foundation of object detection/object regconiztion. Secondly, we discuss a number of object detection techniques found in the literature. Finally, several evaluation metrics for performance comparison on object detection are introduced.

## 2.1. Content Based Image Retrieval

Image retrieval has been an active research area since the 1970's (Rui et al., 1999). Researchers from two different communities, database management and computer vision, proposed two different directions of retrieval, one being text-based and the other visual based. Text-based image retrieval in the past usually required the images to be annotated with words manually before retrieval can be conducted. With the significant advances of database management and textual information retrieval, text-based image retrieval in this framework has achieved some success. However, two major difficulties make this approach unfavorable, especially when a large number of images are involved. The first one is simply the vast amount of labor needed for manual annotation. The second is due to the subjectivity of the annotators; individual persons may perceive images in very different ways, resulting in different labels.

In the early 1990's, Content-Based Image Retrieval (CBIR) emerged as a new technique and started to gain more and more attention. CBIR retrieves images based on the visual content, such as colours and textures, rather than the keywords. Smeulders et al. (2000) gave a thorough overview of CBIR techniques in the literature by the year 2000. Some more recent work on CBIR and also automatic image annotation can be found in the review by Datta et al. (2005). According to the categorization of Cox et al. (2000), image retrieval tasks fall into three different scenarios: category search, target search and association search. In the following, we describe each category in details, together with a discussion of some important relevant work.

### *2.1.1. Category Search*

In category search, the users search for one or more arbitrary image representitives that belong to a prototypical category, such as “cars”, “pedestrians” or “human faces”. Since each category is generally an object class, category search can also be considered as object class recognition.

Early research on object class recognition was mainly focused on face detection (Viola and Jones, 2001), where detection algorithms try to locate human faces in images if there are any, i.e. which image patch is a face. Afterwards, the research area has been extended to generic object detection. For example, Fergus et al. (2003) proposed an “unsupervised scale-invariant learning” approach to discovering object classes including “motorbikes”, “faces”, “airplanes” and “spotted cats” from a data-set containing background images. Objects were modeled as constellations of parts (i.e. salient regions), which were found by a salient region detection algorithm. For each object class, a probabilistic model was estimated on the training set to describe the relationships between the scale, location and appearance of parts. Sivic et al. (2005) and Russell et al. (2006) described the aims of their experiemnts as: “ Is it possible to learn visual object classes simply from looking at images? ”. In other words, they took away the labels from the training set and tried to find object classes purely on the visual information of the images. Unsupervised topic discovery techniques, probabilistic Latent Semantic Analysis (pLSA) (Hofmann, 1999, 2001) and Latent Dirichlet Allocation (LDA) (Blei et al., 2003), were borrowed from the text analysis community. Images were treated as text documents and quantised salient feature descriptors found in images were treated as words. Therefore, discovering object classes from images is analogous to discovering topics from a corpus of text documents.

### *2.1.2. Target Search (Object Retrieve)*

In target search, the users have a precise copy of the image in mind and intended to find the exact same or similar ones from the database. Applications where this type of search is useful include checking if a particular trademark logo has been registered (Eakins et al., 1998), or finding relevant information (e.g. title, artist, etc.) of a specific painting (Hare and Lewis, 2005a). Target search usually resorts to a sample image given by the user, and is known as the “query by example” paradigm. Therefore, rather than asking the system to “find images with trees” as in

the category search scenario, here the users ask it to “find similar images to this one”. Sample images can be images the user already possessed or sketches drawn by the user.

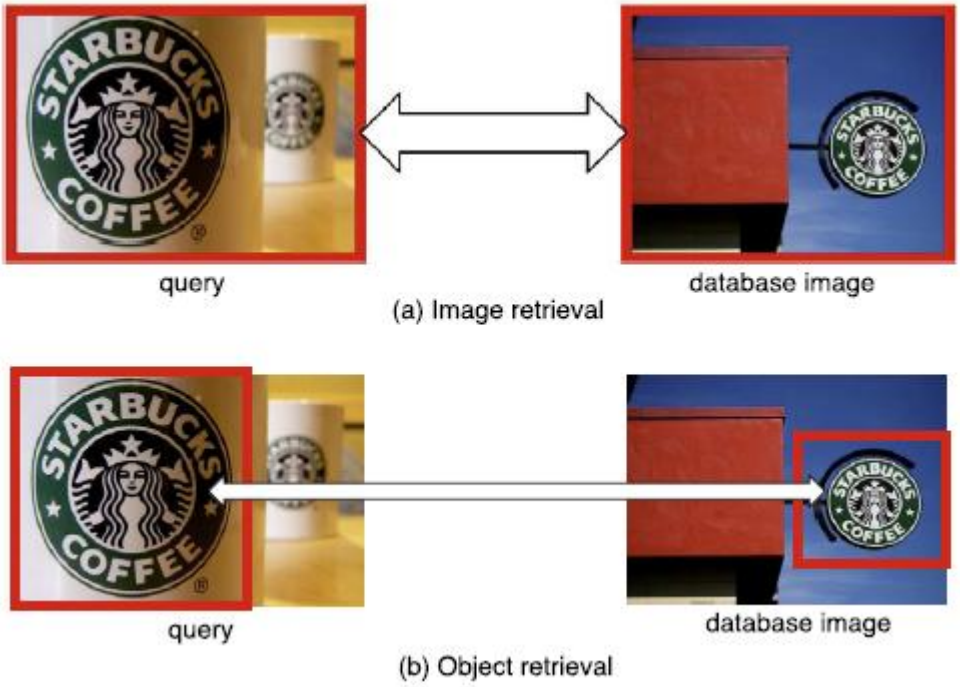



**Figure 2.1 Image retrieval vs. object retrieval.**

**Red rectangles mark the regions of interest of the user. (a) In image retrieval, a user is interested in global appearance of images. The data image is not similar to query image and thus not retrieved. (b) In object retrieval, user is interested in local regions. The data image has a local region that is highly similar to the query object. It should be retrieved. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)**

Jacobs et al. (1995) proposed an approach to retrieving images that are similar to a handdrawn sketch submitted by the user. The similarity between the sketch and each image from the database is calculated as the distance of the coefficients generated by wavelet decomposition of the images. However, in this retrieval mechanism, information about the desired target image can only be expressed to a very limited extent, considering the quality of the sketch. A more general choice is using example images, i.e. an image containing the object or content to be searched for. Numerous researchers have adopted this mechanism in their experiments (Flickner et al., 1995; Mokhtarian et al., 1996; Wang et al., 2001; Hare and Lewis, 2004). Unfortunately, because of the phenomenon which is now commonly referred to as the semantic gap (Smeulders

et al., 2000), this mechanism often fails to capture the similarity that is latent but can be inferred by humans. Besides, finding a suitable example image is a difficult task in itself (Rodden, 1999).

### *2.1.3. Association Search*

Association search is also known as browsing, where users have no particular target image in mind at the beginning of the search. Often, the goal of the search may change, and the users may refine the search through interaction with the system in an iterative manner. Interaction is usually achieved by relevance feedback (Rui et al., 1997; Hiroike et al., 1999) from the user.

A common choice of organising data for browsing is hierarchical structures, which is also known as tree structures. For example, Barnard and Forsyth (2001) proposed a generative hierarchical model for organising images. Both labels and image segments were integrated into the construction of the model. Because of the hierarchical characteristic of the model, image browsing is well supported. An alternative structure is image networks. More flexible navigation is supported in networks, because they need not be acyclic as trees.

### *2.1.4. Image Search in the Real world*


**Figure 2.2 Google Image Search results of (a) “Washington”; (b) “Washington” with “face” constraints**

Google Image Search[1] is probably the most famous real world search engine for images. There are also many similar engines such as Yahoo Image Search, and MSN Image Search, and so on. As many researchers have pointed out, these search engines rely on textual descriptions found on the Web pages containing the images and the file names of the images (Li and Wang, 2006). Although the results of search are fairly good in many cases (e.g. searching for images of a celebrity), one should be aware that the textual descriptions are mostly given by people manually. Considering the effort of manual annotation, the number of images with textual descriptions is probably very small compared with those with no descriptions. In other words, the results are chosen from candidates that constitute only a very small portion of the images actually available on the Internet.

With the advances of research in CBIR, there have been already some realisations of CBIR techniques in real world applications. For example, Google started to integrate face detection techniques into their image search engine. Although the new feature has not been released yet, users can use it by appending "&imgtype=face" to the end of the URL. Figure 2.1 shows the result of a search for images using keyword "Washington" and that of applying "face" constraints. As can be seen, results for "Washington" contains images of maps, places and persons, while that for "Washington" with constraints of "face" filtered out all non-face images. The technique proposed by Jacobs et al. (1995) has been efficiently implemented in Retrievr4, a sketch based image search interface for retrieving images from the online photo sharing website Flickr.

## 2.2. Object detection

In an object retrieval system, as long as images contain the query object, they should be retrieved even though they may globally look quite different from the query image. Such object retrieval techniques though challenging receive strong interests as rising the needs [1,2,4–6] for matching specific images objects in large-scale photo collections subject to occlusions, change of view points, illusions, etc. The system aims to discover similar local regions of images instead of full images. In practice, the user usually tends to provide a query image that depicts only an object. If there are multiple salient objects in the query image provided, the user should mark a

[1] http://www.google.com/imghp?hl=en&tab=wi

region that properly contains the query object that she/he is interested in. The system then returns a ranked list of images that are likely to contain the query object. Due to this, object detection is the most important task of an object retrieval system.

***Definition of object detection:*** *Given an object class of interest T (target, such as pedestrian, human face, buildings, car, horse, bike, text ...) and an image P, object detection is the process to determine whether there are instances of T in P, and if so, return locations where instances of T are found in the image P.*

Most object detection algorithms determine the location of an object based on some characteristic features, i.e., color, edge pattern, size, shape, etc. These algorithms generally are successful when determining the type of object and the location of the object as a whole. Howerver, the main difficulty of object detection arises from high variability in appearance among objects of the same class. An automatic object detection system must be able to determine the presence or absence of objects with different sizes and viewpoints under various lighting conditions and complex background clutters. For a system with the task of locating an object, the algorithm usually groups pixels as either belonging to the object or not, based on some intensity rules, e.g., a neighborhood of pixels that have intensity values within a given range may be grouped together as an object. However, in the situation when an object's position and size are not explicitly known, an algorithm that returns the location of the center of the object, as well as the amount of scaling and rotation, is also useful.

Many approaches have been proposed for object detection in images under cluttered backgrounds. In most approaches, the object detection problem is solved within a statistical learning framework. First, image samples are represented by a set of features, and then learning methods are used to identify objects of interest class. In general, these approaches can be classified as two categories: global appearance-based approaches and component-based approaches.

### *2.2.1. Global appearance-based approaches*

Global appearance-based approaches consider an object as a single unit and perform classification on the features extracted from the entire object. Many statistical learning mechanisms are explored to characterize and identify object patterns. Rowley et al. [7] and Carcia and Delakis [8] use neural network approaches as classification methods in face detection.

Based on wavelet features, Osuna et al. [9] and Papageprgiou and Poggio [10] adopt support vector machines to locate human faces and cars. Schneiderman and Kanade [11] use Naı¨ve Bayes rule for face and non-face classification. Recently, boosting algorithms are applied to detect frontal faces by Viola and Jones [12], then are extended for multi-view face detection by Li et al. [13] and for text detection by Chen and Yuille [14]. Other learning methods used in object detection include probabilistic distribution [15,16], principal components analysis [17] and mixture linear subspaces [18].

### *2.2.2. Component-based approaches*

Component-based methods treat an object as a collection of parts. These methods first extract some object components, and then detect objects by using geometric information. Mohan et al. [19] propose an object detection approach by components. In their approach, a person is represented by components such as head, arms, and legs, and then support vector machine classifiers are used to detect these components and decide whether a person is present. Naquest and Ullman [20] use fragments as features and perform object recognition with informative features and linear classification. Agarwal et al. [21] extract a part vocabulary of side-view cars using an interest operator and learn a Sparse Network of Winnows classifier to detect side-view cars. Fergus et al. [22] and Leibe et al. [23,24] also use interest operators to extract objects' parts and perform detection by probabilistic representation and recognition on many object classes, such as motorbikes, human faces, airplanes, and cars.

### *2.2.3. Discussion*

As opposed to a majority of the above approaches, the problem of detecting multi-class objects and multi-view objects has been recently gained great attention in computer vision community. Schneiderman and Kanade [11] train multiple view-based detectors for profile face detection and car detection. Lin and Liu [25] propose a multi-class boosting approach to directly detect faces of many scenarios, such as multi-view faces, faces under various lighting conditions, and faces with partial occlusions. Amit et al. [26] use a coarse to fine strategy for multi-class shape detection with an application of reading license plates. There are 37 object classes to be recognized, including 26 letters, 10 digits, and 1 special symbol. Li et al. [27,28] propose methods to learn a geometric model of a new object category using a few examples and detect multi-class objects by a Bayesian approach. To improve efficiency, Torralba et al. [29] introduce

an algorithm for sharing features across object classes for multi-class object detection. Tu et al. [30] propose an image parsing framework to combine image segmentation, object detection, and recognition for scene understanding.

One visual task related to object detection is object recognition, whose goal is to identify specific object instances in images. Local descriptor-based methods are increasingly used for object recognition. Schiele [31] proposes to use Gaussian derivatives as local characteristics to create a multidimensional histogram as object representation, and then perform the task to recognize many 3D objects. Lowe [32] develops an object recognition system that uses SIFT descriptors based on local orientation histograms. However, these methods are designed to recognize a specific object rather than in generalization to categorize the object class.

## 2.3. Evaluation of object detection-Precision & Recall

Once the test images are detected by detecting object systems, object detection qualities need to be assessed for performance comparisons between different systems. The common evaluation metrics have been used widely by researchers is *Precision and Recall*.

Precision and recall, which are the most popular metrics for comparing different information retrieval systems, are also widely adopted for evaluating the effectiveness of object detection approaches. In the information retrieval community, precision of a query is defined as the ratio of the number of relevant documents that are returned by the system to the total number of documents returned, and recall is defined as the ratio of the number of relevant documents returned to the total number of relevant documents in the database. Similar, in the object detection community, precision and recall are defined without any different.

Per-image Precision and recall are calculated on the basis of a single test image. For each test image, precision is defined as the ratio of the number of objects that are correctly predicted to the total number of objects predicted, and recall is the ratio of the number of objects that are correctly predicted to the number of objects in the ground-truth image. Mathematically, they are calculated as follows

$$Per\ Image\ Recall = \frac{r}{n} \qquad Per\ Image\ Precision = \frac{r}{(r+w)}$$

where:

r : the number of correctly predicted words;

n : the number of manual labels in the test image;

w : the number of wrongly predicted words.

Per-image precision and recall values are averaged over the whole set of test images to generate the mean/average precision*(AP)* and the mean/average recall*(AR)*.

## 2.4. Summary

In this chapter, we have mainly reviewed the background of this thesis. It began with a brief introduction to the research on CBIR. Three kinds of retrieval scenarios are introduced, namely category search, target search and association search. A few limitations of the traditional CBIR systems were addressed, the semantic gap problem in particular. Then, we went on to discuss object detection techniques, which are relatively new to the image community and seem to be a promising alternative to traiditional CBIR. Two main categories of approaches have been reviewed, Global appearance-based approaches and component based approaches. In the end, we introduced the common performance evaluation metric- precision and recall which are widely used in computer vision field.

# Chapter 3 - Feature Extraction

Digital images are generally considered as two dimensional matrices. Before being analyzed by object detection algorithms, they need to be condensed from pictorial information into feature values, so that the information which is important to the problem being solved can be retained or maximized while the redundancy can be removed. Image description is the process of generating descriptions that represent the visual content of images in a certain manner, normally in the form of one or more features. We view the process of description or feature extraction and feature quantization, as shown in Figure 3.1. Feature extraction condenses pixel values of each interesting region into feature values. Feature quantization maps feature values from continuous space into a discrete space. The first steps exist in most cases, while the second step may be omitted depending on specific approaches. This chapter only focuses on how to extract feature – calculate feature values. The second step is described more detail in next chapter.

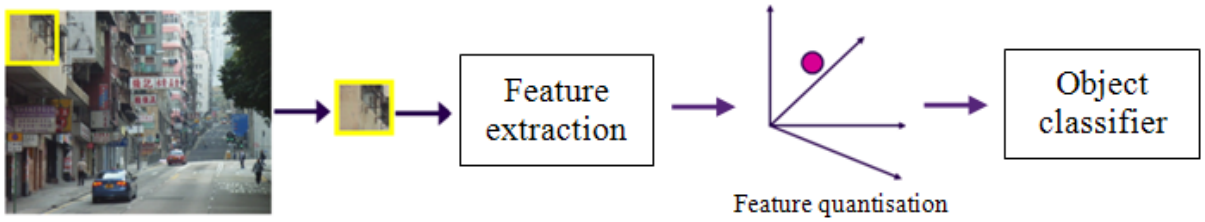



**Figure 3.1 The general process of object detection**

There are two kinds of features global or local. When the interesting region is chosen to be the whole image, features are global, describing the whole image. Other while, in the case the interesting region is chosen to be a partition, segment or salient region, features are local, describing individual parts of the image. In the early years of research on CBIR, global descriptors were the main choices for image description. However, local descriptors have been proposed later as researchers started to realized the limitations of global descriptors, especially for applications where a particular object in the image is of interest. Local description approaches often choose parts from the images firstly, and then calculate descriptors for each individual part. Features can also be categorized as being general or domain-specific. General features include commonly used features such as color, shape and texture. However, for special applications such as fingerprint recognition and lipreading (Matthews et al., 2002), general features are not applicable, so domain-specific features have to be developed. There are a great number of features used by researchers. Previous methods have used many representations for

object feature extraction, such as raw pixel intensities [7,8,33], edges [34], wavelets [9,10,35], rectangle features [12,13,14], and local binary pattern [36]. MPEG-7 (Martinez, 2004) standardized a number of visual descriptors which are selected from many descriptors of a similar kind, through a strict evaluation procedure, and so are recommended as of high performance. Theoretically, the combination of different kinds of features will produce a more robust image description. In the following, some different image features with the details of those used in this thesis are discussed.

## 3.1. Edge/Corner Feature

It is easy to see that the most important features for recognition object are the overall shape of that object and all boundaries which separate main parts of objects.

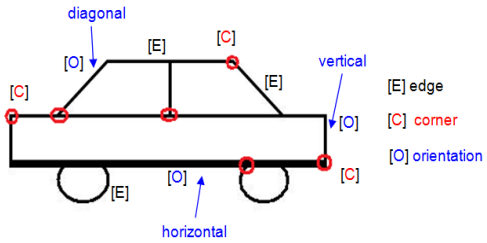



**Figure 3.2 Three main components make object be figured.**

Shape or boundary is basically componented by *edges*. These edges are arranged in an order and meet each other at some points called *corners*. Beside, edge orientation also plays a prominent role for figuring object. With the same number of edges, a talent artist can organize in different order and different orientation to make viewer imagine different objects. Fig. 3.2 describes for these principal components for making object imagination.

### *3.1.1. Edge/Corner Extraction*

Up to present, there are a lot of implementations of edge/corner detection and get the result with a high accuracy rate. Many of them have been reported in public. Zuniga and Haralick fit a continuous surface over a small neighborhood of each point and consider the rate of change in gradient direction. Kitchen and Rosenfeld [97] proposed a cornerness measure based on the change of gradient direction along an edge contour multiplied by the local gradient magnitude. Moravec *et al* defined ”points of interest” as points where there is a large intensity variation in every direction. Harris and Stephens [95] used image derivatives to estimate the

autocorrelation of the image. Rangarajan, Shah and Brackle [94] found an optimal function representing the corner detector which when convolved with the gray level function yields a maximum at the corner point. Rafajlowicz, E. in [98] uses the idea of vertically weighted regression and in its simplest form it leads to interpreting SUSAN in terms of a box sliding on the surface of an image. This modification of the SUSAN algorithm is still simple, robust against errors and provides thinner edges, without further efforts on additional thinning. Canny [96] introduces the notion of non-maximum suppression, which means that given the presmoothing filters, edge points are defined as points where the gradient magnitude assumes a local maximum in the gradient direction. And in 2007, Sonya Coleman *et al* [53] proposed a new method for enabling edge and corner detection to be integrated with a significantly reduced computation time.

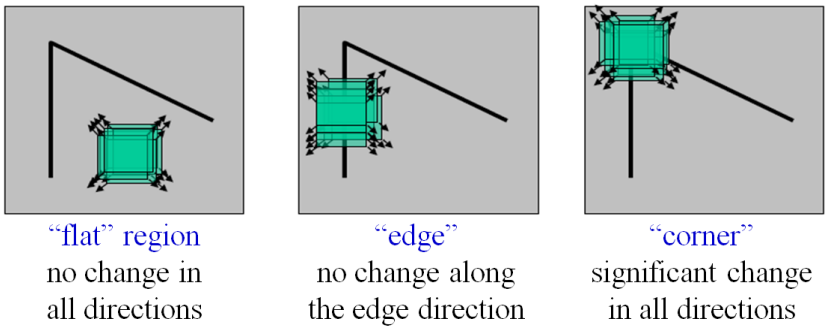



**Figure 3.3 "A Combined Corner and Edge Detector", C.Harris, M.Stephens, 1988**

In our performance, we have tried with many method for edge/corner detecting. And it show that Canny detector gives the best result for edge detection, while Harris detector works well with corner detection.

### *3.1.2. Edge Map and Corner Map*

How do we know that edge/corner is at right position or not? In order to answer this question, we must point out the place where edge/corner must belong to before detecting object. Actually, there are many way to specify location of edge/corner, such as using prior knowledge, or machine learning… In our implement, we choose machine learning approach to make edge map/coner map, which refers the right position of each edge/corner in the target object. This method is more general for a lot of objects and one most important reason is that when using this method, user doesn't have to know more detail about object that they want to detect.

To make edge map and corner map, we use same method as Zhenfeng Zhu *et al* in [40]. For all images in the training-image set, after detecting edge/corner we accumulate them in one temp image $T_e$ or $T_c$. Then edge map and corner map will be made from these temp images respectively. However, [40] makes edge map and corner map be binary images. Our preferment shows that with gray scale image, the result will be better. Because with a binary map, the location of edge or corner must be fixed at that position. But, due to variant of scale or view point, the location of edge or corner can not be fixed at one specific point, other while it can be singed in a small region. So, gray scale map can work better in this case.

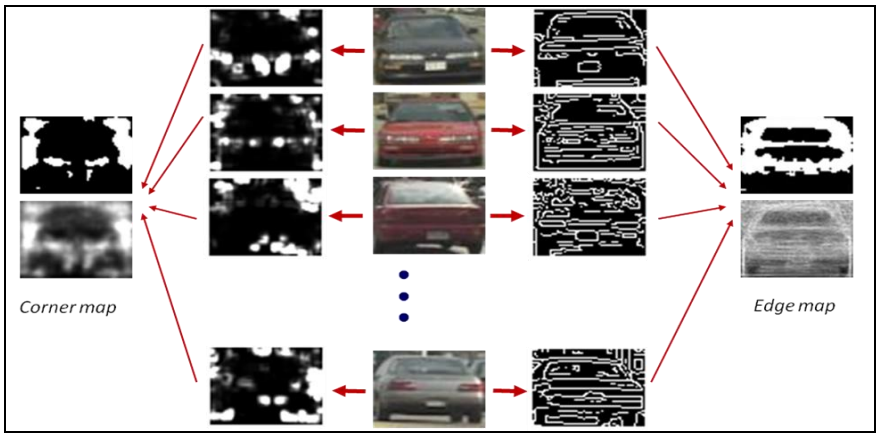



**Figure 3.4 Making Edge map and Corner map from training set.**

The edge map $M_E(x,y)$ and corner map $M_C(x,y)$ are constructed from $T_e$ and $T_c$ as

$$M_E(x,y) = \frac{1}{\theta_1 * N} T_e(x,y), \qquad M_C(x,y) = \frac{1}{\theta_2 * N} T_c(x,y) \qquad (1)$$

where N is the number of images in the training set, and $\theta_1$, $\theta_2$ are specific thresholds. Here, we choose $\theta_1$ equal to 25 and 35 for $\theta_2$. $M_E(x,y)$ and $M_C(x,y)$ denotes for edge map and corner map respectively.

## 3.2. Histogram of Oriented Gradient (HoG)

Specifying orientation of each edge is not an easy task. Because, with all edge detection method above, we only get the total edge image not each separate edge in image. So, it is almost impossible to know the orientation of each edge due to we don't specify each edge individually. Instead of that, we can calculate the domain orientation of all edge in a specific sub-image. From this, we can define relatively the edge orientation. The domain orientation of a region can be

calculated with HoG descriptor method (Histogram of Oriented Gradient). HoG descriptor is a local statistic of the orientations of the image gradients around a keypoint. HoG descriptor was initially proposed by Lowe [54] in his Scale Invariant Feature Transform (SIFT) [54]. Several HoG-based algorithms have been recently presented and combined with technologies such as boosted classifiers for pedestrian detection. SIFT has also been proposed for face recognition [36], though this approach totally differs from ours. In [36] keypoints are located at the local extreme of the scale space as in the original Lowe's approach [54]. Therefore there is no control on the number, position and scale of the keypoints. Navneet Dalal and Bill Triggs (INRIA) have used this method for human detection with a high accuracy rate [56]. David Monzo *et al* [55] compare HoG-EBGM vs. Gabor-EBGM and show the result that HoG has a better performance.

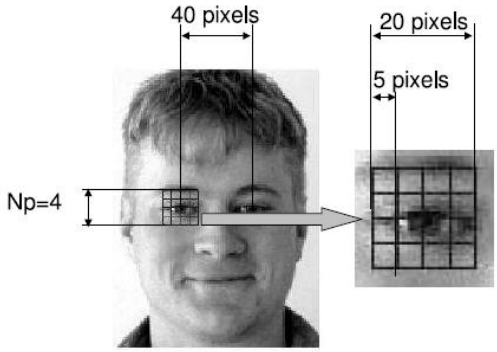



**Figure 3.5 Normalized face and the spatial bins of the right eye HoG descriptor**

The HoG descriptor is a local statistic of the orientations of the image gradients around a keypoint. At each pixel within the detection window the image gradient is computed, and accumulated into bins over (i) orientation, and (ii) spatial regions ('cells'). Within each cell, a histogram of gradient orientation is computed. Although many schemes for dividing the image into cells *e.g*. log-polar schemes have been investigated, using simple square cells is shown to be effective and is computationally efficient. The intuition to binning (quantizing) orientation and spatial position is to introduce some invariance to local image deformation. A second stage of spatial accumulation groups contiguous ranges of cells into 'blocks'. The descriptor for each block is then independently normalized to have constant norm. The blocks typically overlap by one or more cells such that each cell is represented multiple times (with different normalizations) in the final descriptor formed by concatenating all blocks. This gives some contrast invariance over a larger scale than that of the individual cells.

More formally, each HOG descriptor is a histogram in which the bins form a three dimensional lattice with $N_p = 4$ bins for each spatial direction and $N_o = 8$ bins for the orientation for a total of $N_p^2 N_o = 128$ components. In our work, each spatial bin is a $5 \times 5$ pixel square. This size is chosen accordingly to the distance between eyes of the normalized faces, which in our work is 40 pixels. Fig. 3.5 shows the spatial lattice of a HoG descriptor centered on the right eye.

The contribution of each pixel gradient to the histogram is weighted by the gradient modulus and a Gaussian window centered at the keypoint with standard deviation half the extension of the spatial bin range (10 pixels). Also the contribution is trilinearly interpolated with the surrounding bins.

Gaussian windowing and trilinear interpolation increases the robustness of the descriptor against small displacements of the keypoint location. Finally, to increase invariance to non-linear illumination changes HoG descriptors are normalized. See [54] for the minor implementation details. The HoG scheme can be applied to gray-scale images since it makes use of the image gradient alone. However, when applying the descriptor to color images, Dalal & Triggs' implementation computes the gradient at a pixel in each of the red, green and blue channels, and selects the response with greatest magnitude. This can potentially increase the sensitivity to edges in color images which correspond to changes in color with no associated change in luminance. However, because the decision is made independently at every pixel it can also have the effect of increasing noise, and does not enforce any 'coherence' in the edges. Additionally, because the unsigned gradient is typically used, the method is agnostic to dark/light*vs.* light/dark transitions, and again cannot capture the ikelihood that nearby edges are likely to be of the same sign. Due to these disadvantages, in this research, HoG is only calculated on gray scale image.

## 3.3. Circle/Line Feature

Cirlce/Line feature is extremely useful for a specific kind of object that contains strong circular shape or line. For example, with bike in side view, if there is no oclusiion, it must has two wheels. In the case tower object, two lines are along with its body.

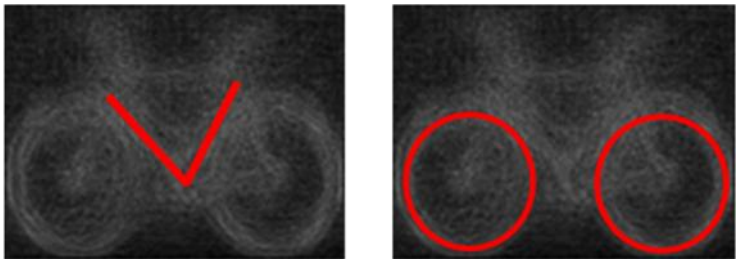


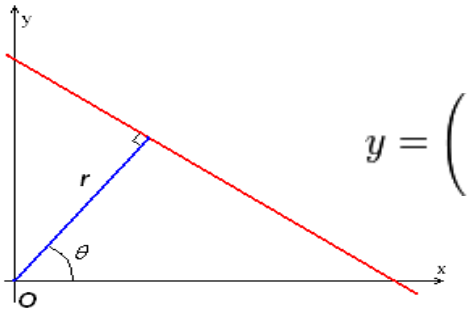
**Figure 3.6 Bycle with strong Line (left) and Cirle (right) feautures**

In this research, the Hough Transfrom is used to detect cirlce/line in image. The classical Hough transform was concerned with the identification of lines in the image, but later the Hough transform has been extended to identifying positions of arbitrary shapes, most commonly circles or ellipses. The Hough transform as it is universally used today was invented by Richard Duda and Peter Hart in 1972, who called it a "generalized Hough transform" after the related 1962 patent of Paul Hough. The transform was popularized in the computer vision community by Dana H. Ballard in 1981.

Due to imperfections in either the image data or the edge detector, there may be missing points or pixels on the desired curves as well as spatial deviations between the ideal line/circle/ellipse and the noisy edge points as they are obtained from the edge detector. For these reasons, it is often non-trivial to group the extracted edge features to an appropriate set of lines, circles or ellipses. The purpose of the Hough transform is to address this problem by making it possible to perform groupings of edge points into object candidates by performing an explicit voting procedure over a set of parameterized image objects (Shapiro and Stockman).

The simplest case of Hough transform is the linear transform for detecting straight lines. In the image space, the straight line can be described as $y = mx + b$ and can be graphically plotted for each pair of image points $(x, y)$. In the Hough transform, a main idea is to consider the characteristics of the straight line not as image points $(x_1, y_1)$, $(x_2, y_2)$, etc., but instead, in terms of its parameters, i.e., the slope parameter $m$ and the intercept parameter $b$. Based on that fact, the straight line $y = mx + b$ can be represented as a point $(b, m)$ in the parameter space. However, one faces the problem that vertical lines give rise to unbounded values of the parameters m and b. For computational reasons, it is therefore better to use a different pair of parameters, denoted $r$ and $\theta$ (*theta*), for the lines in the Hough transform. For an arbitrary point on the image plane with

coordinates, e.g., ($x_0$, $y_0$), the lines that go through it are $r(\theta) = x_0 \cdot \cos\theta + y_0 \cdot \sin\theta$, where $r$ (the distance between the line and the origin) is determined by θ.




$$y = \left(-\frac{\cos\theta}{\sin\theta}\right)x + \left(\frac{r}{\sin\theta}\right)$$

**Figure 3.7 Basis idea of Hough Tranform for detecting line**

This corresponds to a sinusoidal curve in the ($r$, $\theta$) plane, which is unique to that point. If the curves corresponding to two points are superimposed, the location (in the *Hough space*) where they cross corresponds to a line (in the original image space) that passes through both points. More generally, a set of points that form a straight line will produce sinusoids which cross at the parameters for that line. Figure 3.8 presents a visual of Hough space

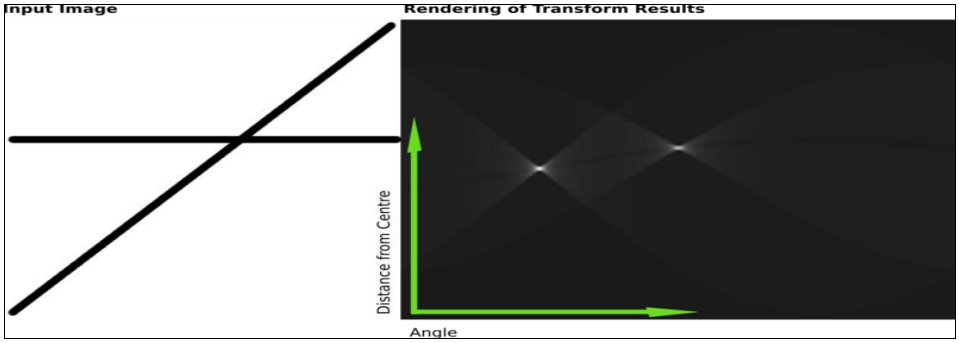



**Figure 3.8 An example of a Hough transform on a raster image containing two thick lines**

The Standard Hough transform (SHT) is a robust technique for detecting any analytically defined shape. But, it requires a considerable amount of memory and has a high computational cost. Due to this, several modified versions have been recognized as robust techniques for curve detection, such as Dynamic Generalized Hough Transform (Henrik et. al., 1995), Transformation of Inversion (TOI), Probobilistic Hough Transfrom (PHT), ... . Among of them, Circular Hough Transfrom (Mohamed Rizon et. al., 2005) shows better perfomance. This method can detect object even polluted by noise. The CHT was sketched by Duda et al. The CHT aims to find circular patterns within an image. The CHT is used to transform a set of feature points in the image space into a set of accumulated votes in a parameter space. Then, for each feature point,

votes are accumulated in an accumulator array for all parameter combinations. The array elements that contain the highest number of votes indicate the presence of the shape. CHT is mainly used for detecting line/cirle in this reseach.

## 3.4. Color Feature

Until now, many features have been dicovered, (edge, corner, cirle, line, HoG). The detection and classification of these features is important for object detection. In general, those image features are mostly detected by differential operators which are commonly restricted to luminance information. However, most of the images recorded today are in color. Therefore, in this section, the focus is on the use of color information to detect and classify object as an image features.

The basic approach to compute color image derivatives is to calculate separately the derivatives of the channels and add them to produce the final color gradient. However, the derivatives of a color edge can be in opposing directions for the separate color channels. Therefore, a summation of the derivatives per channel will discard the correlation between color channels [59]. As a solution to the opposing vector problem, DiZenzo [59] proposes the color tensor, derived from the structure tensor, for the computation of the color gradient. Adaptations of the tensor lead to a variety of local image features, such as circle detectors and curvature estimation [60], [61], [62], [63]. The dichromatic reflection model has been introduced by Shafer [64]. The model describes how photometric changes, such as shadows and specularities, influence the RGB-values in an image. On the basis of this model, algorithms have been proposed which are invariant to different photometric phenomena such as shadows, illumination and specularities [65], [66], [67]. The extension to differential photometric invariance has been proposed by Geusebroek et al. [68]. Van de Weijer et al. [69] propose photometric quasi-invariants which have better noise and stability characteristics compared to existing photometric invariants. But most of these model are high computational cost and take long time for response. In this research, we study on a general method that is easy to implement with a very low cost. That technique is color map.

In order to build a color map, color space takes an important role. The choice of color space can be considered as the primary step in making color map. The RGB color space is the default color space for most available image formats. Any other color space can be obtained

from a linear or non-linear transformation from RGB. The color space transformation is assumed to provide robust parameters against varying illumination conditions. Several color spaces have been proposed and used, such as basic color spaces (RGB), perceptual color spaces (HSI, HSV, HSL, TSL), orthogonal color spaces (YCbCr, YIQ, YUV, YES), perceptually uniform color spaces (CIE-Lab and CIE-Luv). The HSV space is invariant to high intensity at white lights, ambient light and surface orientations relative to the light source and hence, can form a very good choice for skin detection methods.The HSV spacces defines color as Hue—the property of a color that varies in passing from red to green, Saturation—the property of a color that varies in passing from red to pink, Brightness (also called Intensity or Lightness or Value)—the property that varies in passing from black to white. The HSV color space has been used by Brown et al. [70], Garcia and Tziritas [71], McKenna et al. [72], Saxe and Foulds, [73], Sobottka and Pitas [74], Thu et al. [75], Wang and Yuan [76], Zhu et al. [77]. A variation of the HS components using logarithmic transformation, called Fleck HS was introduced by Fleck and has been used by Zarit et al. [78]. Another similar color space is TSL color space which defines color as Tint—hue with white added, Saturation and Lightness. The TSL color space has been used by Terillon et al. [79], Brown et al. [70]. The HSV spaces is also used in this research.

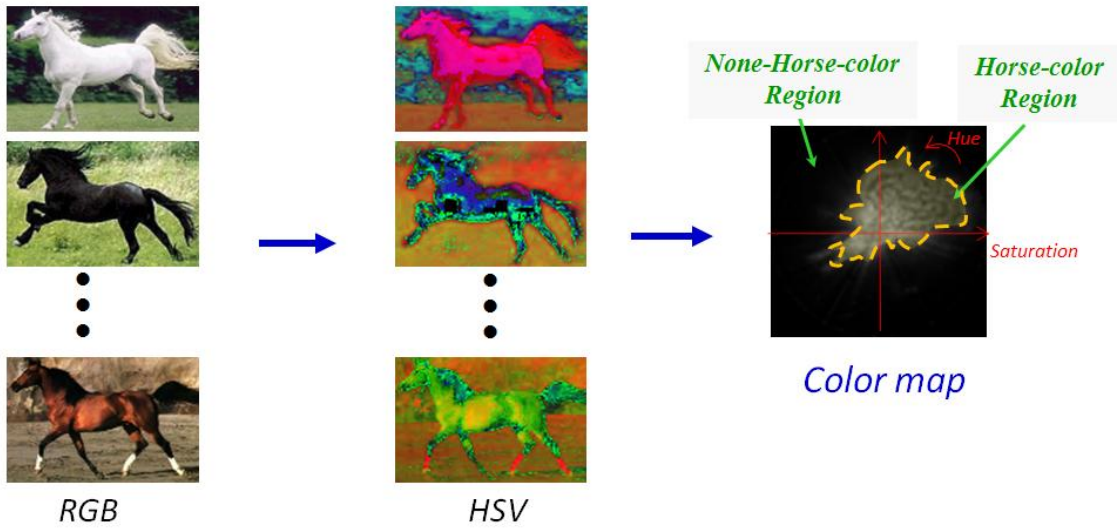



**Figure 3.9 Making Color Map from training image set**

First, every image in the training set is converted from RGB space to HSV space. With each HSV image, at every pixel, we map it value into a square size 255x255 corresponding to two values Hue and Saturation. Then, an average image is taken for all images in training set. Color map, after that, will be done in a two dimensional space (h,s). In this map, usually there are two

parts which refer to region of object-color and non-object-color. Figure 3.9 explains how to get color map from training image. Result of this processing is color map of horse object, which contains two parts: one region of horse color (in the middle of map, inside the yellow boundary) and one region of non-horse color (outside the yellow boundary in black color). After getting color map, how to use it for recognizing object will be discussed more detail in chapter 4.

## 3.5. SURF Feature

In order increase the accuracy rate, after a candidate satisfies edge map, corner map and orientation of edge, that candidate continue to be passed to SURF [39] checking stage. We choose SURF because it is invariant with scale, illumination, and similar but faster than SIFT. This section will describe how to extract SURF feature in an image.

SURF (Speeded-Up Robust Features) is a novel detector-descriptor scheme, coined SURF (Speeded-Up Robust Features). The detector is based on the Hessian matrix [81], but uses a very basic approximation, just as DoG [54] is a very basic Laplacian-based detector. It relies on integral images to reduce the computation time and therefore called the 'Fast-Hessian' detector. Much of the performance increase in SURF can be attributed to the use of an intermediate image representation known as the "Integral Image" [80]. The integral image is computed rapidly from an input image and is used to speed up the calculation of any upright rectangular area. Given an input image I and a point (x; y) the integral image IP is calculated by the sum of the values between the point and the origin.

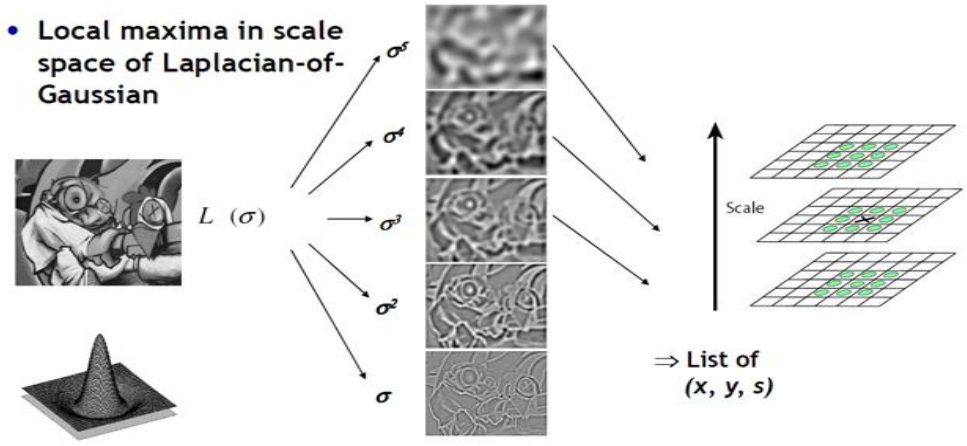



**Figure 3.10 Detect SURF interest points**

After that, SURF detector is used to find interest points based on the determinant of the Hessian matrix. In order to detect interest points using the determinant of Hessian, it needs to find extrema across all possible scales through scale-space. The scale-space is typically implemented as an image pyramid where the input image is iteratively convolved with Gaussian kernel and repeatedly sub-sampled (reduced in size). Figure 3.10 shows the general ideas of SURF detector to find interest points. More detail can be found in [39].

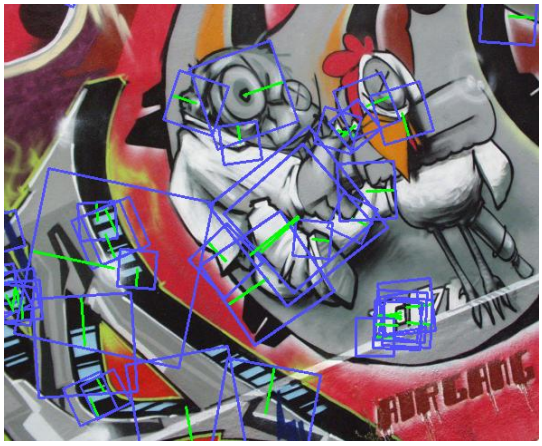


**Figure 3.11 Descriptor Windows. The window size is 20 times the scale of the detected point and is oriented along the dominant direction shown in green.**

After finding interest points, Interest Point Descriptor is calculated. The first step in extracting the SURF descriptor is to construct a square window around the interest point. This window contains the pixels which will form entries in the descriptor vector and is of size 20σ, again where σ refers to the detected scale. Furthermore the window is oriented along the dominant direction, such that all subsequent calculations are relative to this direction.

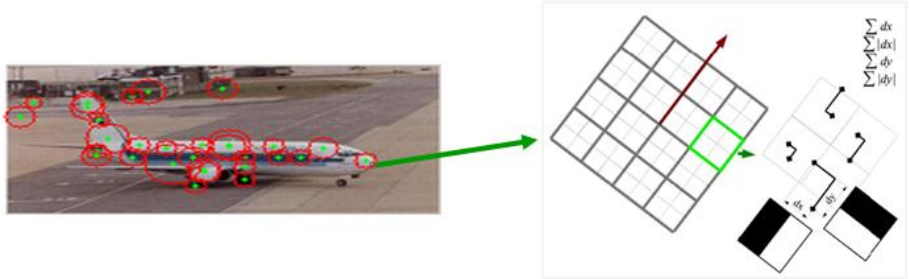


**Figure 3.12 Descriptor Components. The green square bounds one of the 16 subregions representing the sample points at which we compute the wavelet responses. As illustrated the x and y responses are calculated relative to the dominant orientation.**

The descriptor window is divided into 4x4 regular subregions. Within each of these subregions Haar wavelets of size 2σ are calculated for 25 regularly distributed sample points. If we refer to the x and y wavelet responses by dx and dy respectively then for these 25 sample points (i.e. each subregion) we collect,

$$v_{subregion} = \left[\sum dx, \sum dy, \sum |dx|, \sum |dy|\right].$$

Therefore each subregion contributes four values to the descriptor vector leading to an overall vector of length 4x4x4 = 64. The resulting SURF descriptor is invariant to rotation, scale, brightness and, after reduction to unit length, contrast.

## 3.6. Summary

We have described the main aspect of image description, feature extraction. Depend on the interesting region chosen for extracting feature, it can be global feature (if whole image is chosen) or local feature (or part-based feature corresponding to a part of image). Both global and local feature are used for description image. Global feature describes the context information of that image, while local feature presents more detail about some small regions. Due to that, combination both of them often gets better result. There are a great number of different feature descriptors, from basic edge, corner, circle, line, color, descriptors to more advanced descriptors such as HoG, SURF. In some cases, the quantization step is adopted to group feature descriptors into classes, so that CBIR or object detection systems can avoid dealing with a massive number of feature descriptors which are possibly of high dimension.

# Chapter 4 - Object Detection Based on Multi Features

Object detection is a process of identifying and locating objects in a scene. It is a process of combining digital image processing and computer vision (Gonzalez *et al*, 2001). The main goal of an object detection process is to detect and identify objects in a scene. Selected parameters, which are commonly used for object detection such as the interpretation parameters aimed to recognizes and identify objects, is described in the previous section. This part only focuses on how classifier learns to know if an input image is an instance of object or not. We use both local and global features to improve performance.

## 4.1. Why Multi Features?

The problem of detecting and classifying regions and objects in images is a challenging task due to ambiguities in the appearance of the visual data. These ambiguities may arise either due to the physical conditions such as illumination and pose of the scene components with respect to the camera, or due to the intrinsic nature of the data itself. The use of context can help alleviate this problem significantly.

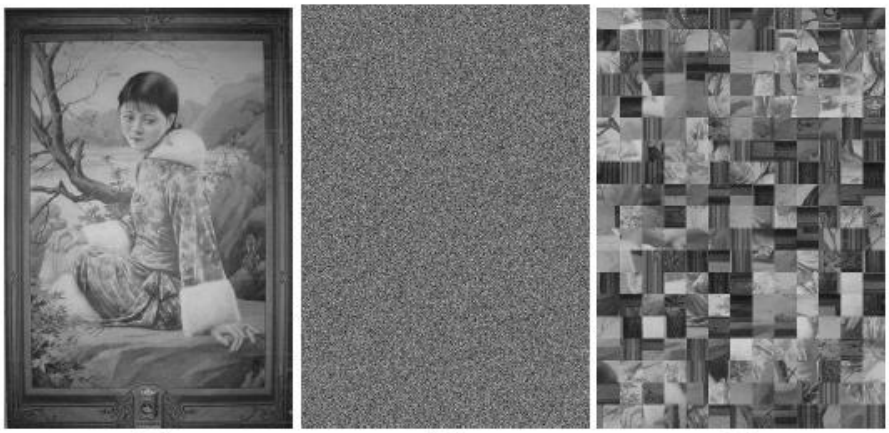



**Figure 4.1 An illustration of the fact that natural images contain strong spatial dependencies rather than being a bag of random independent pixels or blocks. (a) A natural image. (b) Image obtained by randomly scrambling the pixel intensities of the original image in (a). (c) Image obtained by randomly scrambling the original image blocks.**

For example, as shown in Figure 4.1, just on the basis of the appearance, it may be difficult to differentiate a sky patch from a water patch but their relative spatial cconfiguration

with respect to other regions removes this ambiguity. Similarly, a patch from a tree may appear locally very similar to another patch from a building. But if we look at larger neighborhoods of the patch, it is easy to classify which patch is a building patch.

It is well known that natural images are not a random collection of independent pixels. To illustrate this point, a natural image is shown in Figure 4.1 (a). Figure 4.1 (b) shows the image obtained by randomly scrambling the pixels of the previous image. It is obvious that the original image gives us a perception of a coherent scene because there are spatial dependencies in the image which are lost in the scrambled image. The scrambled image seems like random noise even though all the intensities, present in the original image, are also present in this image. Similarly, if one now scrambles bigger blocks instead of pixels (Figure 4.1 (c)), the coherency is still broken.

This demonstrates that it is important to use not only one feature for the analysis of natural images, instead of that, multi features would be better in providing "meaning" of that image. In fact, one would like to have total freedom in modeling long range complex data interactions without restricting oneself to small local nneighborhoods. This idea forms the core of the proposed research in this thesis. The spatial dependencies may vary from being local to global and the challenge is how to combine both global and local features in the best way for detecting an instant of object.

The next section present our approach for detecting an instant of object in a still image by combination many features from local to global one such as edge feature, corner feature, HoG feature, circle feature, line feature, color feature, and SURF feature. How to extract and represent these features is describe detail in the previous chapter.

## 4.2. Object detection approach

In computer vision community, object detection has been a very challenging research topic. Given an object class of interest T (target, such as pedestrian, human face, buildings, car or text) and an image P, object detection is the process to determine whether there are instances of T in P, and if so, return locations where instances of T are found in the image P. The main difficulty of object detection arises from high variability in appearance among objects of the same class. An automatic object detection system must be able to determine the presence or absence of objects with different sizes and viewpoints under various lighting conditions and

complex background clutters. Many approaches have been proposed for object detection in images under cluttered backgrounds. In most approaches, the object detection problem is solved within a statistical learning framework. First, image samples are represented by a set of features, and then learning methods are used to identify objects of interest class. In general, these approaches can be classified as two categories: global appearance-based approaches and component-based approaches.

Global appearance-based approaches consider an object as a single unit and perform classification on the features extracted from the entire object. Many statistical learning mechanisms are explored to characterize and identify object patterns. Rowley et al. [7] and Carcia and Delakis [8] use neural network approaches as classification methods in face detection. Based on wavelet features, Osuna et al. [9] and Papageprgiou and Poggio [10] adopt support vector machines to locate human faces and cars. Schneiderman and Kanade [11] use Naı¨ve Bayes rule for face and non-face classification. Recently, boosting algorithms are applied to detect frontal faces by Viola and Jones [12], then are extended for multi-view face detection by Li et al. [13] and for text detection by Chen and Yuille [14]. Other learning methods used in object detection include probabilistic distribution [15,16], principal components analysis [17] and mixture linear subspaces [18].

Component-based methods treat an object as a collection of parts. These methods first extract some object components, and then detect objects by using geometric information. Mohan et al. [19] propose an object detection approach by components. In their approach, a person is represented by components such as head, arms, and legs, and then support vector machine classifiers are used to detect these components and decide whether a person is present. Naquest and Ullman [20] use fragments as features and perform object recognition with informative features and linear classification. Agarwal. et al. [21] extract a part vocabulary of side-view cars using an interest operator and learn a Sparse Network of Winnows classifier to detect side-view cars. Fergus et al. [22] and Leibe et al. [23,24] also use interest operators to extract objects' parts and perform detection by probabilistic representation and recognition on many object classes, such as motorbikes, human faces, airplanes, and cars.

As opposed to a majority of the above approaches, the problem of detecting multi-class objects and multi-view objects has been recently gained great attention in computer vision community. Schneiderman and Kanade [11] train multiple view-based detectors for profile face

detection and car detection. Lin and Liu [25] propose a multi-class boosting approach to directly detect faces of many scenarios, such as multi-view faces, faces under various lighting conditions, and faces with partial occlusions. Amit et al. [26] use a coarse to fine strategy for multi-class shape detection with an application of reading license plates. There are 37 object classes to be recognized, including 26 letters, 10 digits, and 1 special symbol. Li et al. [27, 28] propose methods to learn a geometric model of a new object category using a few examples and detect multi-class objects by a Bayesian approach. To improve efficiency, Torralba et al. [29] introduce an algorithm for sharing features across object classes for multi-class object detection. Tu et al. [30] propose an image parsing framework to combine image segmentation, object detection, and recognition for scene understanding.

One visual task related to object detection is object recognition, whose goal is to identify specific object instances in images. Local descriptor-based methods are increasingly used for object recognition. Schiele [31] proposes to use Gaussian derivatives as local characteristics to create a multidimensional histogram as object representation, and then perform the task to recognize many 3D objects. Lowe [32] develops an object recognition system that uses SIFT descriptors based on local orientation histograms. However, these methods are designed to recognize a specific object rather than in generalization to categorize the object class. This part proposes a generic approach for detecting object by combination both global and local features.

### *4.2.1. General Approach for Object Detection*

Based on image representation through features, we present an object detection approach using a combination of both local and global features in this research. Our approach uses a hierarchical object detector combining many techniques to detect objects, and learns informative features for the classifier. Unlike methods which use interest operators to detect parts prior to recognition of the object class, we apply the proposed object detector at anywhere in image scale space. Therefore, our method does not need figure-ground segmentation or object parts localization. In contrast to most systems which are designed to detect a single object class, our method can be applied to any type of object classes with widely varying texture patterns and varying spatial configurations. Extensive experiments on eight different kinds of objects (side view car, front/rear view car, bike, motorbike, train, plane, horse and sheep) are conducted to evaluate the proposed object detection approach.

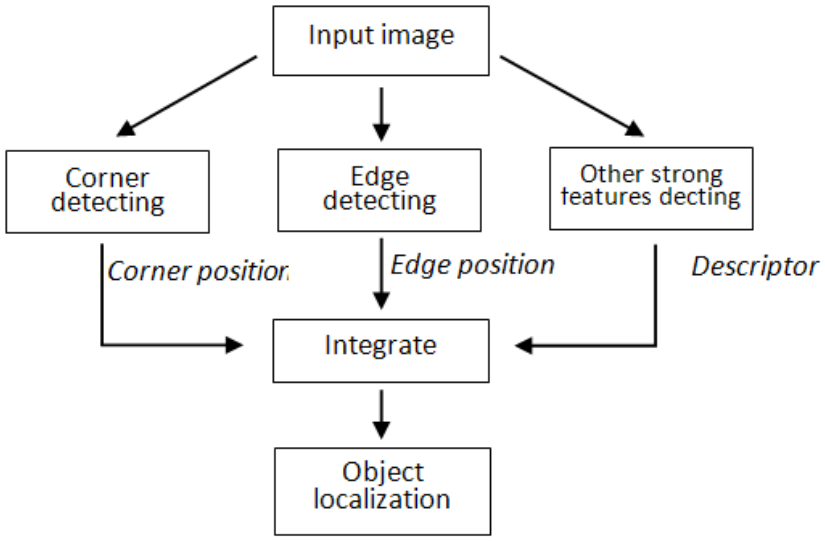



**Figure 4.2 General proposed approach for object detecting based on multi features**

First, we employ feature extract technique to find and disciple local and global features in the input image. This is presented detail in chapter 3. After getting feature descriptions, each of them will be consequently passed to each step of our approach to check if it satisfies condition of an object instant or not. If all of them satisfy then it would be a candidate of object. Other while, only one feature description doesn't satisfy, it is not an instant of object. The rest of this part would show more particularly how to know it satisfy or not.

### *4.2.2. Satisfy Edge Map*

As denote as in chapter 3, the edge map and corner map are constructed from $T_e$/$T_c$ as

$$M_E(x,y) = \frac{1}{\theta_1 * N} T_e(x,y), \qquad M_C(x,y) = \frac{1}{\theta_2 * N} T_c(x,y) \quad (1)$$

where N is the number of images in the training set, and $\theta_1$, $\theta_2$ are thresholds. Here, we choose $\theta_1$to equal 25 and $\theta_2$ to equal 35. $M_E(x,y)$ and $M_C(x,y)$ denotes for edge map and corner map respectively.

With an input image, we will make edge image- $I_E$ and corner image-$I_C$. To extract edges for making edge image, we use canny edge detector with 3x3 structure. And to consider the position of corners, Harris corner detector is used with 5x5 structure. Beside, Gaussian filter is applied on input image before extracting edge and corner features to subtract background.

Edge image after detecting edge from input one is denoted as $I_E$. Let's define two parameters:

$n_1^e$ : number of edges in the input edge image $I_E$.

$n_2^e$ : number of edges matching between $I_E$ and $M_E$.

Degree of satisfying $M_E$ of $I_E$ is determined by some conditions as bellow:

$$C_1^E(I_E) = \begin{cases} 1 & n_2^e > N_1^e \\ 0 & else \end{cases} \quad (2)$$

$$C_2^E(I_E) = \begin{cases} 1 & \frac{n_2^e}{n_1^e} > \theta^e \\ 0 & else \end{cases} \qquad C_3^E(I_E) = \begin{cases} 1 & n_1^e < N_2^e \\ 0 & else \end{cases} \quad (3)$$

where $N_1^e$, $N_2^e$ are constants and $\theta^e$ is a given threshold. $C_1^E(I_E)$ guaranties that edges in the input image must be in correct position to construct object like edges in $M_E$. While both $C_2^E(I_E)$ and $C_3^E(I_E)$ is to avoid situation that there are so many edges in the input image. So, if $Q_E = C_1^E(I_E) \cap C_2^E(I_E) \cap C_3^E(I_E)$ (4) is true, then $I_E$ satisfies $M_E$, and the image will be passed to the next corner test step.

But, as we mention in part two, after detecting edge we cannot know exactly individual edge. It also means that we cannot calculate $n_1^e$ and $n_2^e$. However, instead of counting number of edges, we can sum of pixels which are marked as edge. Thus, $n_1^e$ and $n_2^e$ can be known approximately as

$$n_1^e \approx \frac{1}{255}\sum I_E(x,y) \qquad n_2^e \approx \frac{1}{255\times 255}\sum I_E(x,y)\cdot M_E(x,y) \quad (5)$$

here, because both $I_E$ and $M_E$ are gray scale, we should divide the sum to 255 for each image.

### *4.2.3. Satisfy Corner Map*

Similarity to the edge test step, in this stage we also define three conditions:

$$C_1^C(I_C) = \begin{cases} 1 & n_2^c > N_1^c \\ 0 & else \end{cases} \quad (6)$$

$$C_2^C(I_C) = \begin{cases} 1 & \frac{n_2^c}{n_1^c} > \theta^c \\ 0 & else \end{cases} \qquad C_3^C(I_C) = \begin{cases} 1 & n_1^c < N_2^c \\ 0 & else \end{cases} \quad (7)$$

where $N_1^c$, $N_2^c$ are constants, $\theta^c$ is a given threshold, $n_1^c$ is the number of corners in the input corner image $I_C$ and $n_2^c$ is the number of corner matching between $I_C$ and the corner map image $M_C$.

$$n_1^c \approx \frac{1}{255}\sum I_C(x,y) \qquad n_2^c \approx \frac{1}{255\times 255}\sum I_C(x,y)\cdot M_C(x,y) \quad (8)$$

then, qualification of whether the input image satisfies corner map or not is

$$Q_C = C_1^C(I_C) \cap C_2^C(I_C) \cap C_3^C(I_C) \quad (9)$$

### *4.2.4. Satisfy Histogram of Oriented Gradient (HoG)*

From the input image I(x,y), we find the orientation of each pixel by using HOG descriptor method as in chapter 3. The magnitude m(x, y) and the orientation θ(x, y) are computed by

$$\begin{aligned} m(x,y) &= \sqrt{g_x(x,y)^2 + g_y(x,y)^2} \\ \theta(x,y) &= \tan^{-1}(g_y(x,y)/g_x(x,y)) \end{aligned} \quad (10)$$

where $g_x(x, y)$ and $g_y(x, y)$ denotes the x and y components of the image gradient in x and y direction calculated by 1-D centered mask [−1, 0, 1]

$$g_x(x, y) = I(x + 1, y) - I(x - 1, y) \quad (11)$$

$$g_y(x, y) = I(x + 1, y) - I(x - 1, y)$$

After that, we specify some sub regions of object where the orientation is so strong such as horizontal, vertical and diagonal.

**Figure 4.3 Sub regions with strong orientation. From left to right: edge image, horizontal, diagonal and vertical of car object.**

The orientation is ranged from [0 − 2$\pi$], we divide into 16 bins: *h[0]*... *h[15]*. For each sub regions, we calculate histogram of orientation in that region by quantizing the orientation $\theta(x,y)$ for all pixels falling to *h[i]* bins, $i = \overline{0,15}$, weighted by its magnitude $m(x, y)$.

Condition for input image satisfies orientaiton is that

$$H(S) = \begin{cases} 1 & \dfrac{h[0]+h[15]+h[7]+h[8]}{\sum_{i=0}^{15} h[i]} > \sigma_1 \\ 0 & else \end{cases} \quad (12)$$

$$V(S) = \begin{cases} 1 & \dfrac{h[3]+h[4]+h[11]+h[12]}{\sum_{i=0}^{15} h[i]} > \sigma_2 \\ 0 & else \end{cases} \quad (13)$$

$$D(S) = \begin{cases} 1 & \dfrac{h[1]+h[2]+h[9]+h[10]}{\sum_{i=0}^{15} h[i]} > \sigma_3 \\ 0 & else \end{cases} \quad (14)$$

where $\sigma_1$, $\sigma_2$, and $\sigma_3$ are specific thresholds. If all seven sub-regions satisfy condition of the strong orientation respectively then input image satisfies orientation.

### *4.2.5. Satisfy Color Map*

In the section 3.4, technique to extract color feature and build up color has been presented. This part focus on how to know an input image satisfies color map or not. Figure 4.4 shows two basic steps for working with color feature.

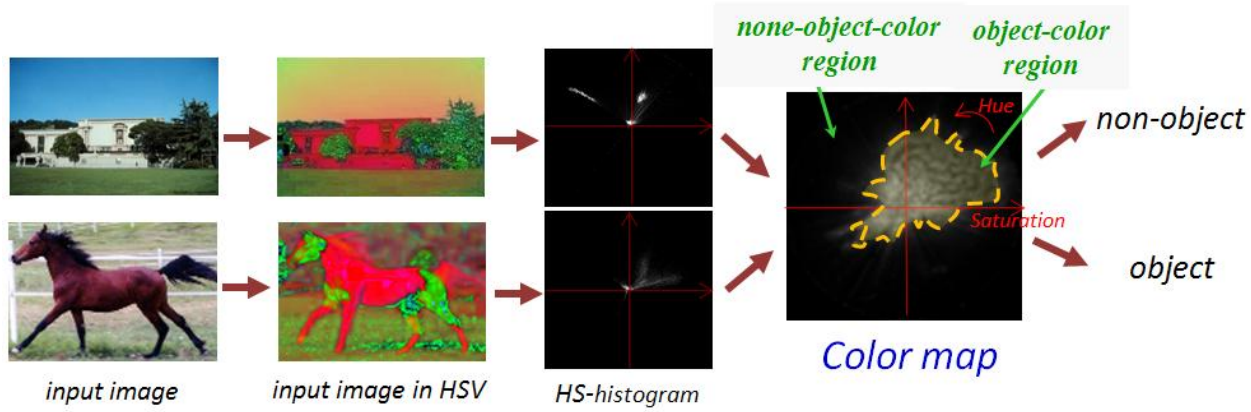



**Figure 4.4 Diagram of checking color map**

First, input image is converted from RGB to HSV space. After that, histogram of H(hue) and S(saturation) will be calculate. HS-histogram is then compared with color map of object to see which region it falls into. If it is in the object-color region, that is an instant of object. Constrastly, it is considered a non-object image in the case it falls into non-object region.

Let's denote CM(x,y) for color-map image and H(x,y) for HS-histogram image of input image. Color matching is calculated by

$$\mathrm{M}_{\mathrm{color}} = \sum_{x,y} CM(x,y) * H(x,y) \qquad (15)$$

Originally, $\mathrm{M}_{\mathrm{color}}$ is a matching-score which refers how many pixel of HS-histogram has the same value with that pixel in color map correspondly. But, both HS-histogram and color map are gray scale image, not binary image, $\mathrm{M}_{\mathrm{color}}$ may present for degree of HS-histogram that are recognized as color of object. Due to this, if $\mathrm{M}_{\mathrm{color}}$ is over a threshold then we can say that input image satisfies color, other while, it is not instant of object. Condition of color satisfaction is as bellow:

$$\mathrm{M}_{\mathrm{color}} > \delta \qquad (16)$$

where $\delta$ is a threshold we have to define before detecting. Value of $\delta$ can be learned from training image set and varies depend on what kind of object we want to detect.

### *4.2.6. Satisfy SURF*

In order increase the accuracy rate, after a candidate satisfies edge map, corner map and orientation of edge, that candidate continue to be passed to SURF [39] checking stage. We choose SURF because it is invariant with scale, illumination, and similar but faster than SIFT.


**Figure 4.5 Making vocabulary for matching SURF feature**

Method of matching between two set of SURF descriptors as Gabriella *et al* in [57, 58] is used. After describing all SURF features as in section 3.6, bag of keypoints or vocabulary are made by quantizing all these descriptors from training image set to K bins as in figure 4.5. Training image set includes both positive and negative set. Value of K is an integer depending on what kind of SURF descriptor we used. In this research, because we use 64-SURF, so K can varies from 50 to 1000. The larger K is, the more time system takes for learning and for detecting. But the smaller K is, the worse system's performance for detecting is. Our experiments show that K is about 100 or 500 get the best result.

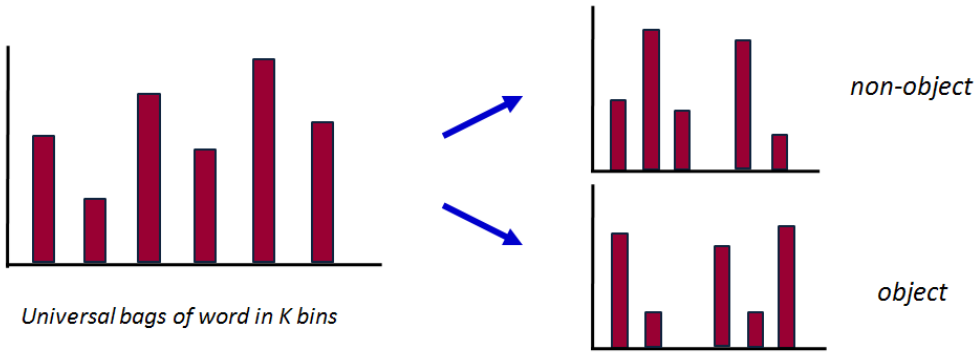



**Figure 4.6 Quantize to form bag of words vector for object/non-object**

After quantizing all 64-SURF descriptors into K bin, we get the universal bags of word. Based on this, from positive training set and negative training set, object or non-object histogram of SURF descriptors will be made as in figure 4.6.

Once descriptors have been assigned to form feature vectors for object/non-object, with a new input image, we use Naïve Bayes classifier to determine whether this input image belongs to one category or not by taking the largest posterior score, $\text{argmax}\{P(C_j \mid I)\}$

$$P(C_j \mid I) \quad \alpha \quad P(C_j)P(I \mid C_j) = P(C_j)\prod_{t=1}^{k} N^{(t,I)} P(v_t \mid C_j) \qquad (17)$$

where I is new image, $C_j$ is category class (object/non-object), $v_t$ represents for keypoint or cluster center, and $N^{(t,I)}$ is the number of times that keypoint $v_t$ occurs in image I.

## 4.3. Results of object detection

This section gives detailed experiments and discussions on quality issues of our proposed method for object detecting. We have applied our detection method to eight kinds of object: side view car, front/rear view car, bike, motorbike, train, plane, horse and sheep. Through the experiments and by comparisons of the results, we have a good result.

This part is based the poster presentation publication by the author in (Bao and Shirai, 2010) at IEICE-ISS conference in Sendai, 2010. It begins with a description of our experimental image datasets. Afterwards, it shows the results and discussion of object detection.

### *4.3.1. Image Database*

Our proposed scheme is evaluated on eight object-categories, including front/rear view car, side view car, bicycle, motorbike, train, plane, horse and sheep.

**Figure 4.7 Training and testing images of side- (left) and front/rear- (right) view car**

We use CALTECH-256[2] image databased for testing with front/rear view car, bicyle, motorbike and airplane. While side view car database is downloaded from UIUC[3] webpage.

[2] http://www.vision.caltech.edu/html-files/archive.html

[3] http://cogcomp.cs.illinois.edu/Data/Car/

Horse object is download form Weizmann[4] webpage. With train, horse and sheep object, we have to select manualy from internet. Beside, in order to evaluate the result more exactly, we also get object images from the officcial website of PASCAL 2010[5].

Morover, an additional about 1500 negative examples are collected randomly from many websites and added to the database. Thus with each object-category, the whole database contains about more than 1500 negative images and 1500 positive images (including 1000 images for training, and 500 images for testing). Some examples of our image database including training and testing images are presented in figure 4.7 to 4.10.

**Figure 4.8 Training and testing image of bike (left) and train (right)**

**Figure 4.9 Training and testing image of plane (left) and motorbike (right)**

[4] http://www.wisdom.weizmann.ac.il/~vision/databases.html

[5] http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2010/index.html

Image databases contain natural images that are taken from several sources and include occlusion and cluttered backgrounds. With side view car, images used for training have the same size 40x100 as in original database. Otherwhile, all images in the training set of front/rear view car, we resize manually into 60x80 to eliminate the influence from background. The basic size of bicycle image is set to 40x80, motorbike is 100x150, horse is 85x130, sheep is 100x130 and, airplane is 50x150.

**Figure 4.10 Training and testing image of horse (left) and sheep (right)**

Train object is specially, because train has no specific size. The length of train object changes very much. This is one of our obstacle when testing with train object. Because if train object has no fix-size, then it can't make Edge map and Corner map. The safest way we choose is that trying to detect one coach in stead of whole train. After detecting some coachs, the post-processing will be done to make the whole train from these coachs. Size of one coach is 40x150. Detecting results are presented in next part.

### *4.3.2. Experimental results*

With a testing image, we preprocess it by resizing it into four times as size of training image. After detecting at this scale, we reduce it size and continue detecting until it's size is equal to training image. This work guarantees to be able to detect all objects in the input image even if the size of object in it is smaller or larger than basic size of training object. After detecting we have a list of candidates for an object. Every candidate has a "*satisfied score*", which refers how much that candidate can be an instant of object. Based on this score, we intergrate some of candidates into a real instant of object if these candidates are too close. Other while, if "satisfied score" is too low, that candidate will be eliminated automactically. Summary

of detection for every kind of object is listed in table, afterthat, some of experimental results for each object-category are shown in following figure, from figure 4.11 to 4.18.

| Test | No. image | Correct | Incorrect | False rate | Accuracy rate |
|---|---|---|---|---|---|
| Caltech | 504 | 481 | 23 | 4.57% | 95.43% |
| Rear Caltech-09 | 516 | 475 | 41 | 8.95% | 92.05% |
| Non- car 1 | 165 | 155 | 10 | 6.06% | 93.94% |
| Non – car 2 | 380 | 350 | 30 | 7.89% | 92.11% |

**Table 4.1 Result of front/rear view car detection**


**Figure 4.11 Example of front/rear view car detection result**

| Test | No. image | Correct | Incorrect | False rate | Accuracy rate |
|---|---|---|---|---|---|
| UIUC1 | 300 | 275 | 25 | 8.33% | 91.67% |
| UIUC Test | 278 | 264 | 14 | 5.04% | 94.96% |
| Non- car 1 | 500 | 480 | 20 | 4.00% | 96.00% |
| Non – car 2 | 380 | 362 | 18 | 4.74% | 95.26% |

**Table 4.2 Result of side view car detection**

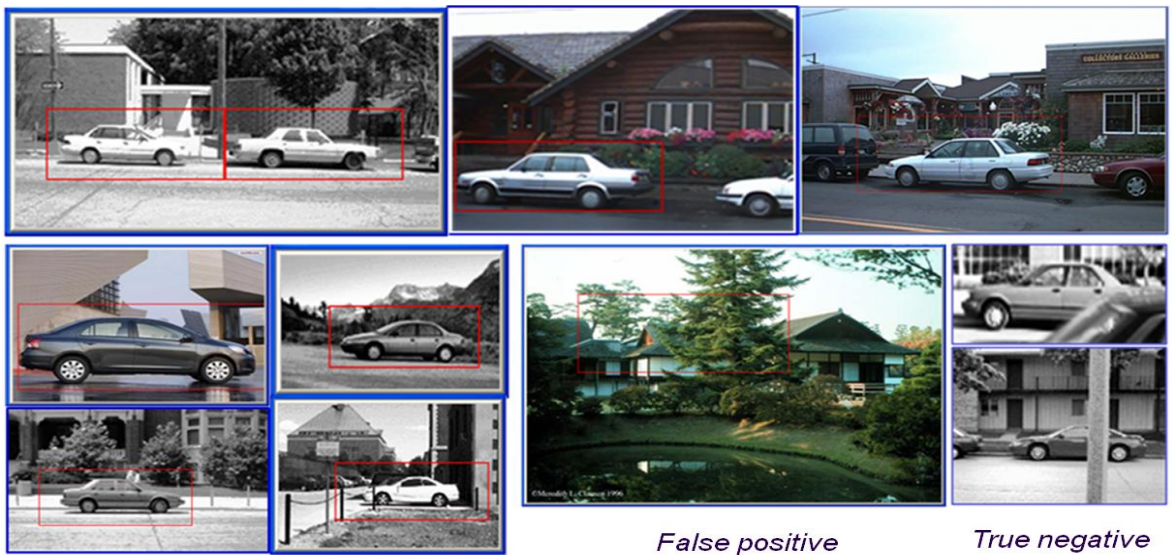



**Figure 4.12 Example of side view car detection result**

| Test | No. image | Correct | Incorrect | False rate | Accuracy rate |
|---|---|---|---|---|---|
| Caltech bike | 365 | 261 | 104 | 28.49% | 71.51% |
| PASCAL10 | 387 | 295 | 92 | 23.77% | 76.23% |
| Non- bike 1 | 171 | 140 | 31 | 18.13% | 81.87% |
| Non- bike 2 | 380 | 356 | 24 | 6.32% | 93.68% |

**Table 4.3 Result of bicycle detection**


**Figure 4.13 Example of bicycle detection result**

| Test | No. image | Correct | Incorrect | False rate | Accuracy rate |
|---|---|---|---|---|---|
| PASCAL 10 | 378 | 300 | 78 | 20.63% | 79.37% |
| Web | 395 | 306 | 89 | 22.53% | 77.47% |
| Non- train 1 | 165 | 135 | 30 | 18.18% | 81.82% |
| Non- train 2 | 380 | 345 | 35 | 9.21% | 90.79% |

**Table 4.4 Result of rail train detection**


**Figure 4.14 Example of rail train detection result**

| Test | No. image | Correct | Incorrect | False rate | Accuracy rate |
|---|---|---|---|---|---|
| Caltech | 677 | 594 | 83 | 12.26% | 87.74% |
| Caltech 09 | 987 | 803 | 184 | 18.64% | 81.36% |
| Non- plane 1 | 245 | 208 | 37 | 15.17% | 84.83% |
| Non- plane 2 | 380 | 340 | 40 | 10.53% | 89.47% |

**Table 4.5 Result of plan detection**

**Figure 4.15 Example of plane detection result**

| Test | No. image | Correct | Incorrect | False rate | Accuracy rate |
|---|---|---|---|---|---|
| Caltech 256 | 270 | 188 | 82 | 30.37% | 69.63% |
| VOC 2010 | 296 | 205 | 91 | 30.74% | 69.26% |
| Weizmann | 328 | 271 | 57 | 17.38 | 82.62% |
| Non- horse 1 | 245 | 194 | 51 | 20.82% | 79.18% |
| Non- horse 2 | 380 | 326 | 54 | 14.21% | 85.79% |

**Table 4.6 Result of horse detection**


**Figure 4.16 Example of horse detection result**

While testing with sheep object we make a modification of how to match edge map ($M_e$)/corner map($M_c$) with edge image ($I_E$)/corner image($I_C$). Previous version uses 0x0=0, it means that if there is no edge/corner pixel in $I_E/I_C$ image, then that pixel is considered as a wrong position no matter what there is or there is not edge/coner pixel in $M_e/M_c$ map or not. We see that it is not good. Because in the case there is no edge/conrer in $M_e/M_c$ map, if there is no edge/corner pixel in $I_E/I_C$ image, then that pixel is actually at right position. Due to this, when working with sheep, the 0x0=0.5 method is used instead of 0x0=0. Figure 4.17 descrbiles more detail about this idea.

| $M_c/M_e$ | $I_C/I_E$ | Matching |
|---|---|---|
| 0 | 0 | 0.5 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

**Table 4.7 Matching score**

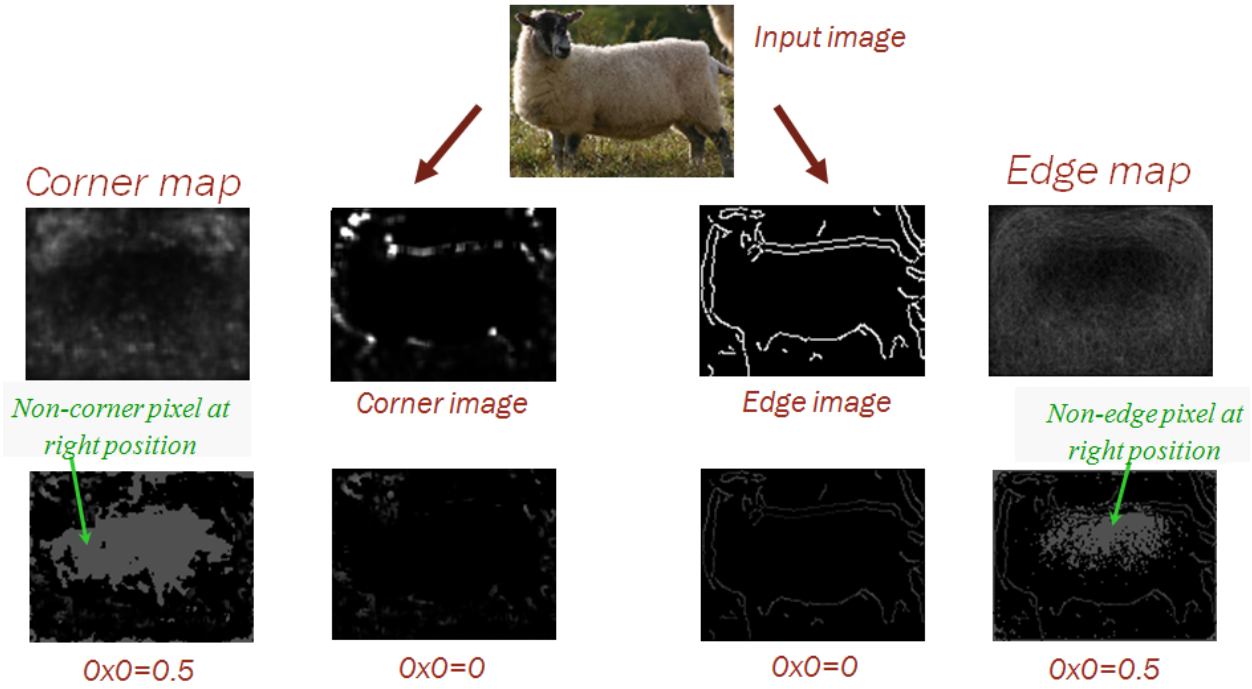



**Figure 4.17 Visualization of two matching methods: 0x0=0 and 0x0=0.5**

| Test | No. image | OLD METHOD (0x0=0) | | | | UPDATED METHOD (0x0=0.5) | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | True | False | False rate | Accuracy rate | True | False | False rate | Accuracy rate |
| **VOC 2010** | 576 | 363 | 213 | 36.98% | 63.02% | 406 | 170 | 29.51% | 70.49% |
| **Internet** | 831 | 565 | 266 | 32.01% | 67.99% | 606 | 225 | 27.08% | 72.92% |
| **Non-sheep 1** | 245 | 152 | 93 | 37.96% | 62.04% | 196 | 49 | 20.00% | 80.00% |
| **Non-sheep 2** | 380 | 243 | 137 | 36.25% | 63.95% | 299 | 81 | 21.32% | 78.68% |

**Table 4.8 Sheep detection result with two methods 0x0=0 and 0x0=0.5**

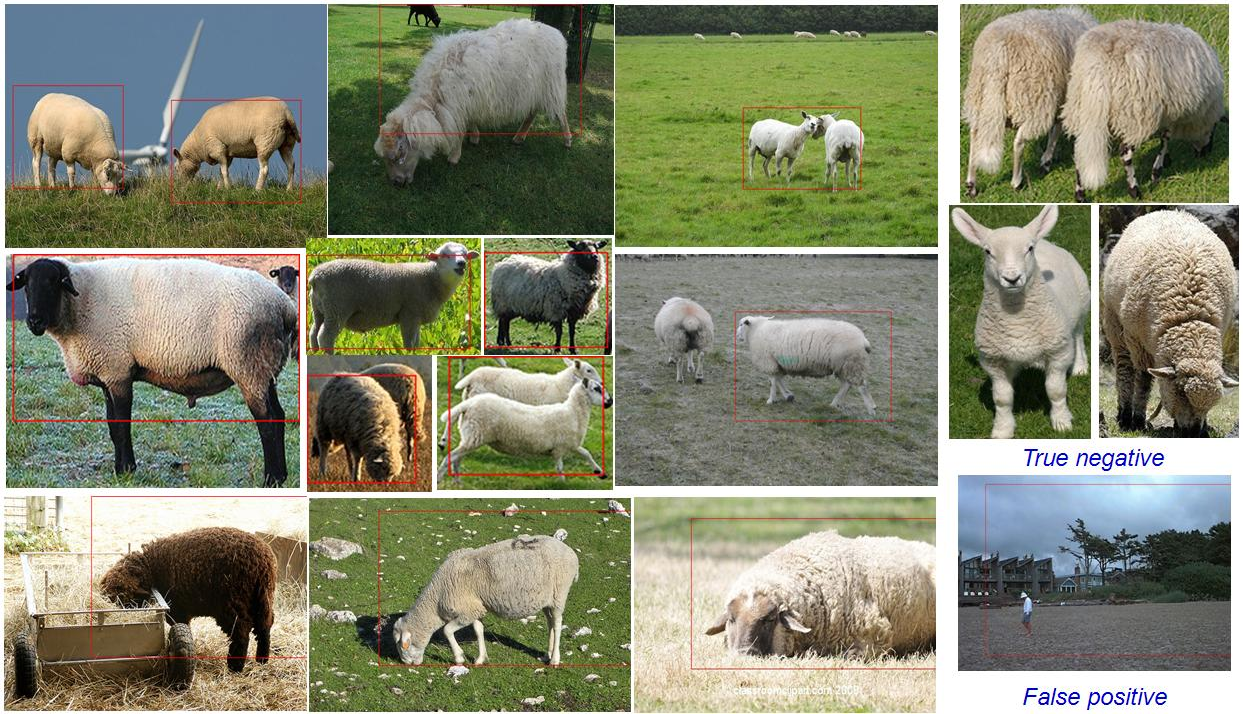



**Figure 4.18 Example of sheep detection result**

Table 4.9 shows result with eight categories of object. It presents that in generally new matching method (0x0=0.5) gets better perfomance than old method (0x0=0).

| Object | OLD METHOD (0x0=0) | | UPDATED METHOD (0x0=0.5) | |
|---|---|---|---|---|
| | Average Precision | Average Recall | Average Precision | Average Recall |
| **F/r car** | 97.40% | 89.71% | **97.45%** | **89.77%** |
| **Side car** | 95.83% | 92.38% | **95.91%** | **92.43%** |
| **Bike** | 83.76% | 72.64% | **83.82%** | **80.69%** |
| **Train** | 84.05% | 74.10% | 81.57% | 76.43% |
| **Aero plane** | **85.59%** | **87.56%** | 81.15% | 86.54% |
| **Motorbike** | **95.63%** | **85.35%** | 95.35% | 84.47% |
| **Horse** | 88.64% | 68.78% | **88.37%** | **70.72%** |
| **Sheep** | 74.84% | 65.96% | **86.13%** | **71.93%** |

**Table 4.9 Comparison between old and new matching method for all objects**

In special case, if there are many non-edge/non-corner pixels are inside object (for example bicycle, sheep or horse object), then new matching method (0x0=0.5) will hence better result. In constract, the old matching method (0x0=0) is suitable with the case that many non-edge/non-corner pixels are outside of object (motorbike, aeroplane). With train object, because

there are little non-edge/non-corner pixels in edge map ($M_e$)/corner map ($M_c$), both two matching methods 0x0=0 and 0x0=0.5 do the same function, and the result is not different so much. This is visualized in figure 4.19.

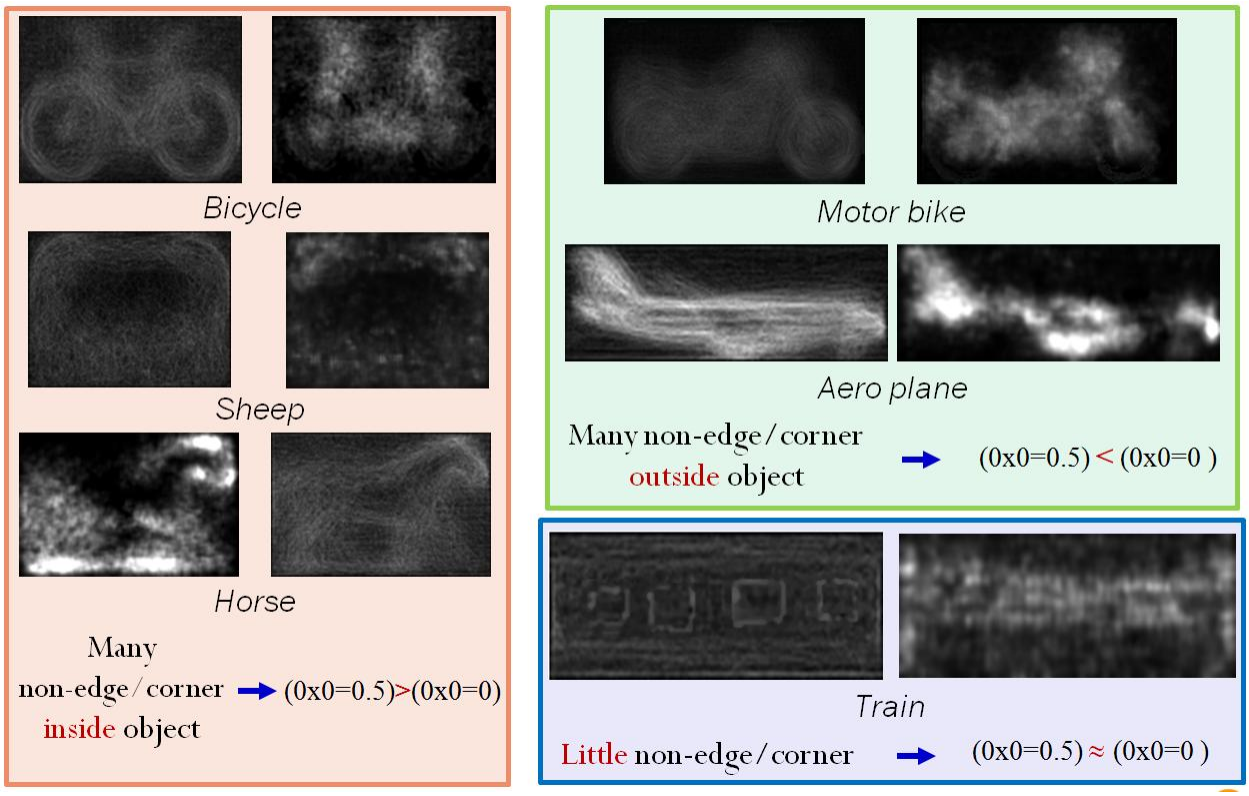



**Figure 4.19 Performance of matching method depends on kind of object**

## 4.4. Evaluation and summary

Taking into account that both local and global features may provide useful information to recognize object, the combination of multi features to construct a system gets good performance. We have presented a general feature-based object detection approach with combination of many features such as edge, corner, HoG, SURF and color. This method selects features, and learns a hierarchical classifier by combining cascade classifiers to detect objects in images. Extensive object detection experiments show high detection rates with relatively low numbers of false detections. These results illustrate the high discriminant power of combination both local and global features and the effectiveness and robustness of the hierarchical object detection approach. The proposed approach is able to detect objects that consist of distinguishable parts in spatial configurations not only natural objects such as horse, sheep; but also man-made objects such as side-view car, front/rear view car, bicycle, motorbike, train and aeroplane. In summary, the

results show that the object representation using combination multi features is general to different kinds of object classes, and our feature selection methods are efficient to extract informative class specific features for object detection.

We evaluate a large image database and the outcome is satisfactory. Compared with PASCAL 2010 result, the consequence of our research is better. Table 4.10 shows comparison between our results with PASCAL 2010 outcome. This result is satisfactory.

| Object | Our system | PASCAL VOC 2010 (5) | |
|---|---|---|---|
| | Average Precision | Average Precision | Authors |
| Front/rear car | 97.45% | 49.10% | UOCTTI_LSVM_MDPM |
| Side car | 95.91% | | |
| Bicycle | 83.82% | 55.30% | NLPR_HOGLBP_MC_LCEGCHLC |
| Train | 84.05% | 50.30% | MITUCLA_HIERARCHY |
| Aero plane | 85.59% | 58.40% | UVA_GROUPLOC |
| Motorbike | 95.63% | 56.30% | NUS_HOGLBP_CTX_CLS_RESCORE_V2 |
| Horse | 88.37% | 51.90% | NUS_HOGLBP_CTX_CLS_RESCORE_V2 |
| Sheep | 86.13% | 37.80% | UVA_DETMONKEY |

**Table 4.10 Our performance compares with PASCAL VOC 2010[6]**

However, there are some restrictions. If the input image is blurring or object is occluded, it cannot be detected. If there are edges and corners arranged similar to the object, our schema knows that it is the object to detect. Between specific size object (car, bike ...) and unspecific size object (train ...), detecting result of the unchanged-size object is better. In table 2, train detecting result is the lowest, because train has no specific size. It can not make properly Edge map and Corner map. But when combining with SURF, it gets better result. SURF feature works well with object having "large surface" (plane, train). In contrast, for example with bike object, almost SURF features fall into background and do not belong to object. With such object, SURF feature is not good. Color feature is only suitable for natural objects with small change in color such as sheep, horse, or animal. Beside, in our proposal method, we have to define features before detecting, but every object has a power-feature to recognize. This is a disadvantage because it is not easy to know which feature is good for an object. Next chapter will discuss how to automatically select strong feature for a specific kind of object.

[6] http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2010/index.html

# Chapter 5 - Automatic Feature Selection

The previous work on chapter 3 and chapter 4 has emphasized the issues of feature extraction and classification for object detection, however, relatively less attention has been given to the critical issue of feature selection. The main trend in feature extraction has been representing the data in a lower dimensional space, for example, using principal component analysis (PCA). Without using an effective scheme to select an appropriate set of features in this space, however, these methods rely mostly on powerful classification algorithms to deal with redundant and irrelevant features. In this chapter, we argue that feature selection is an important problem in object detection and demonstrate an algorithm providing a simple, general, and powerful framework for selecting good subsets of features, leading to improved detection rates. The goal is searching in the feature space to select a subset that plays an important role for detecting the target object of interest. We have tested the proposed framework on ten object detection applications: eight as in chapter 4, plus two other challenging objects in both natural and manmade (flower object and tower object). Our experimental results illustrate significant performance improvements in most cases.

## 5.1. Necessary of automatic feature selection

For any recognition task, it is important to be able to extract features from an image. While computer vision texts will throw the term around as if it's a new vocabulary term, in meaning it is the same as the standard use of the word. Even when a human looks at an object, they see features of an object. Features may include whether the object is round, what size it is, and what color it is. These features essentially describe the object, for example a small, round, red object describes an apple, and a long, curved, yellow object describes a banana. So a technique for object recognition can be to identify and extract features from an input and process these features to determine what sort of object they describe. As these features describe objects, they are often called descriptors. After computing features, it may then be possible to use them to find objects in an image for which features can again be computed, resulting in object features. Most of the state-of-the-art approaches for object detecting often define features before detecting. However, manual identification features is at best time consuming and usually optimistic. Because of that, firstly, feature identification need to be accurate, repeatable and quantitative.

Secondly, it is difficult to figure out which feature is good for a type of object. Due to that, the main advantage of this research is that our system can automatically choose features which are best for detecting one type of object.

Choosing an appropriate set of features is critical when designing pattern classification systems under the framework of supervised learning. Often, a large number of features are extracted to better represent the target concept. Without employing some kind of feature selection strategy, however, many of them could be either redundant or even irrelevant to the classi1cation task. It is possible to make two arbitrary patterns similar by encoding them with a sufficiently large number of redundant features. As a result, the classifier might not be able to generalize nicely. Ideally, we would like to use only features having high separability power while ignoring or paying less attention to the rest. For instance, in order to allow a bicycle detector to generalize nicely, it would be necessary to recognize two wheels (circle feature) encoding fine details which might appear in most of side view bike. A limited yet salient feature set can simplify both the pattern representation and the classifiers that are built on the selected representation. Consequently, the resulting classifier will be more efficient.

In most practical cases, relevant features are not known a priori. Finding out what features to use in a classification task is referred to as feature selection. Although there has been a great deal of work in machine learning and related areas to address this issue these results have not been fully explored or exploited in emerging computer vision applications. Only recently there has been an increased interest in deploying feature selection in applications such as face detection [81,82], gender classification [83,84], vehicle detection [81,85], image fusion for face recognition [86], target detection [87], pedestrian detection [88], tracking [89], image retrieval [90], and video categorization [91].

In this research, we propose using probability method to search the space of features with the goal of selecting a subset of features encoding important information about the target concept of interest. The proposed approach has the advantage that it is simple, general, and powerful. An earlier version of this work has appeared in Bao and Shirai (PRMU, 2011, BKC, Japan) and it relates to our previous work on eight object detection, however, the size of the classes considered here is larger with two other object: tower and flower. In section 5.2, we review the problem of feature selection, emphasizing different search and evaluation strategies. An overview of the proposed method is presented in section 5.3. After that, experimental results and comparison for

detecting ten kind of object are presented in section 5.4. Comparison between automatic feature selection and manual selection is showed in section 5.5. Finally, section 5.56 presents summary and our conclusion.

## 5.2. Feature Selection Problem

The feature selection problem involves the selection of a subset of d features from a total of N features, based on a given optimization criterion. Let us denote the N features uniquely by distinct numbers from 1 to N, so that the total set of N features can be written as $F = \{f_1, f_2, \ldots, f_n\}$. $F_C$ denotes the subset of selected features and $F_N$ denotes the set of remaining features. So, at any time $F = F_C \cup F_N$. Function $g(F_C)$ denotes a function evaluating the performance of $F_C$. Function $g(F_C)$ may evaluate either the accuracy of a specific classifier on a specific data set (e.g., the wrapper approach) or a generic statistical measurement (e.g., the filter approach). The choice of evaluation function $g(F_C)$ depends on the particular application. Formally, the problem of feature selection can be described as bellow:

***Definition of feature selection problem***: With a given feature set F, find a subset $F_C$ such that

$$g(F_C) \quad = \quad \underset{\forall F_S \subseteq F}{\arg\max} \; (g(F_S))$$

The general approach for feature selection problem is exhaustive search. It guarantees to find the optimal result according to the evaluation criterion used. While an exhaustive search is complete (i.e., no optimal subset is missed), a search does not have to be exhaustive in order to guarantee completeness. The exhaustive approach to this problem would require examining all $C_n^d$ possible *d*-subsets of the feature set $F_C$. The number of possibilities grows exponentially, making exhaustive search impractical for even moderate values of *n*. Cover and Van Campenhout [91] showed that no nonexhaustive sequential feature selection procedure can be guaranteed to produce the optimal subset. They further showed that any ordering of the error probabilities of each of the $2^n$ feature subsets is possible. Different heuristic functions can be used to reduce the search space without jeopardizing the chances of finding the optimal result. Hence, although the order of the search space is $O(2^n)$, a smaller number of subsets are evaluated. In this paper, we introduce an approach based on probability method. Since we are maximizing $g(F_S)$, one possible criterion function is the probability of "*satisfy*". The use of probability as a criterion function makes feature selection more generic and not dependent on the specific

classifier used and the size of the training and test data sets. More detail about this approach is discussed in next section.

## 5.3. Algorithm for automatic feature selection

We introduce the basic operations that allow the search toward local optima during the feature selection process. In describing these operations, we do not show explicit parameters for the size of F and $F_C$ since size of F may vary depending on how may features we use, and size of $F_C$ will be defined clearly after we run algorithm. Before describing detail about our proposed algorithm, we should clarify some technique words used in this section.

### *5.3.1. Definition of "Satisfy Feature"*

The training image set should contain both positive images ($\mathbf{T_{pos}}$) and negative images ($\mathbf{T_{neg}}$). Each positive image should display an object of the target category. Negative images may display anything except any objects of the target category.

With a given image *i*, value of feature *f* in that image *i*, is written as ***val(f,i)***.

***Definition 1***: With a given image *i*, one feature *f* is called "***satisfy"***, written ***sat(f,i)***, if value of feature *f* in image *i* falls into range of all values of positive images ($T_{pos}$)

$$\textbf{\textit{sat(f,i)}} \quad : \quad \min_{i_{pos} \in T_{pos}}(val(f, i_{pos})) \le val(f, i) \le \max_{i_{pos} \in T_{pos}}(val(f, i_{pos})) \qquad (18)$$

***Definition 2***: A given image *i* is called *"**satisfy**"* a feature set F, written ***sat(F,i),*** if image *i* ***"satisfy"*** all features of set F

$$\textbf{\textit{sat(F,i)}} \ : \ \forall f \in F, \ sat(f,i) \qquad (19)$$

***Definition 3***: With a given image set I and a specific feature set F, ***sat(F,I)*** is a set of image, member of T that ***"satisfy"*** all features of feature set F

$$\textbf{\textit{sat(F | I)}} \ = \ \{i \in I, \ sat(F,i)\} \qquad (20)$$

These definitions will be used more common in our algorithm from this time to the end. Actually, in an algorithm for automatic feature selection, there are three main things that must make clear: how we initialize the chosen feature set $F_C$ (initial step); how we define the function

g(F) to calculate the fitness value (evaluation step); and when the system will stop (ending step). In the next section, we discuss more detail about three main steps in our algorithm.

### *5.3.2. Best-fixed algorithm for automatic feature selection*

In this section, we derive the feature evaluation function and describe its incorporation into the entire selection procedure.

The following items constitute the input to the selection procedure:

- An initial feature set, $F = \{f_1, f_2, \ldots, f_n\}$.
- A training image set, $T = T_{pos} \cup T_{neg.}$

The selection procedure returns the set of selected features, $F = \{f^1, f^2, \ldots, f^k\}$. The number k is not given before running algorithm. The algorithm will deal with the problem of seeking the optimal number of selected features that depends on what kind of object. Our experiment results show that number of k usually 4 or 5, rarely smaller than 3 or larger than 6. The selection procedure uses the evaluation module and therefore makes certain assumptions regarding its functioning. Figure 5.1 displays a schematic overview of the entire selection module.

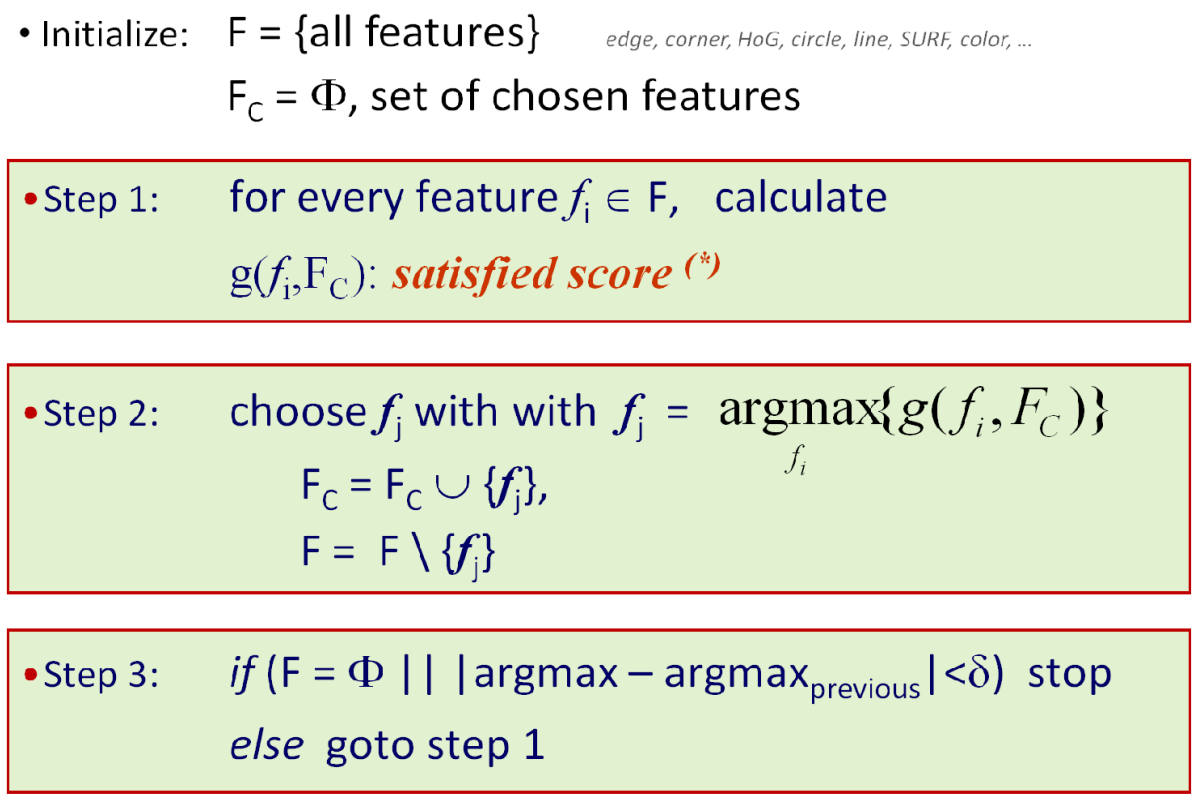



**Figure 5.1 Best fixed algorithm for automatically choosing features**

The basic idea of this algorithm is "best-fixed" approach. From the set of feature, we calculate the score for every feature, and then we choose the one with highest score. The chosen feature will be put in $F_C$ (set of chosen features). After that, we will again estimate the satisfy-score for every feature left. This time the satisfy-score bases on combination of every feature left with all features of $F_C$. It means that all the best features of the previous running are fixed for this running time. It will be looped until there is no any feature left or until we get a good result (the argmax at this running time is smaller than the previous satisfied score). According to our experiment, it is difficult to run out of feature set F, we should stop when it reaches a peak of satisfied score.

***Initial step***

How to initialize chosen feature set $F_C$ depends on search strategy. There are two kind of initialization: sequential forward selection (SFS) and sequential backward selection (SBS). Both SFS and SBS are two well-known heuristic algorithms in feature selection schemes. SFS, starting with an empty feature set, selects the best single feature and then adds that feature to the feature set. SBS starts with the entire feature set and at each step drops the feature whose absence least decreases the performance. In this research, we choose SFS because of its efficiency in running time. Beside, computation for every feature always gets better performance than calculation for many features.

***Evaluation step***

This section gradually derives the feature evaluation function, $g : F_S \rightarrow R$, which assigns ***satisfied score*** to a subset features $F_S$ from feature set F. To begin with, let us define two other sub functions that play important role for calculate satisfied score of g

$$pos : F_S \rightarrow R$$

$$pos(\{f_i\} \cup F_C) = count(T_{pos}| F_C \cup \{f_i\})*1/|T_{pos}| \qquad (21)$$

$$neg : F_S \rightarrow R$$

$$neg(\{f_i\} \cup F_C) = (|T_{neg}| - count(T_{neg}| F_C \cup \{f_i\}))*1/|T_{neg}| \qquad (22)$$

with, count(T|A) is how many images of image set T that satisfy all features of feature set A, it means count(T|A) = | ***sat(A | T)*** | (see more detail in equation 20) . And |T| denotes

cardinality of set T. Not loss generality, pos($\{f_i\} \cup F_C$) shows *percent of positive training set* ***satisfies*** *($f_i \wedge F_C$)*; otherwhile, neg($\{f_i\} \cup F_C$) presents for *percent of negative training set* ***does not satisfy*** *($f_i \wedge F_C$)*.

Satisfied score g is computed as follow:

$$pos : F_S \rightarrow R$$

$$g(\{f_i\} \cup F_C) = \omega_p * pos(\{f_i\} \cup F_C) + \omega_n * neg(\{f_i\} \cup F_C) \quad (21)$$

bias $\omega_p$ and $\omega_n$ are weight of positive and negative score in the total satisfied score.

***Ending step***

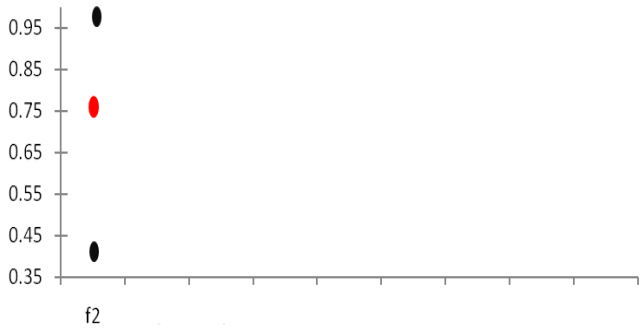
A stopping criterion determines when the feature selection process should stop. Some frequently used stopping criteria are the search completes, some given bound is reached, where a bound can be a specified number (minimum number of features or maximum number of iterations), subsequent addition (or deletion) of any feature does not produce a better subset; a sufficiently good subset is selected (e.g., a subset may be sufficiently good if its classification error rate is less than the allowable error rate for a given task). In this research, we combine two conditions for ending our algorithm: when the search completes (means that F is null) or when adding new feature to $F_C$ makes new $F_C$ becomes worse (means that satisfied score at this running time is smaller than satisfied score of the previous running time).

### *5.3.3. Visualization for the best-fixed algorithm for automatic feature selection*

This section presents an example for visualization for the proposed method for automatic feature selection. Supposed that there are eight features in the F, denotes F = $\{f_1, f_2, \ldots, f_8\}$. Chosen feature set $F_C=\Phi$ at the initialize step. The proposal algorithm can be seen as follow:

***Running step 1***: calculate satisfied score for every features by using evaluation function *g($f_i$)*, with *i*={1,..,8}

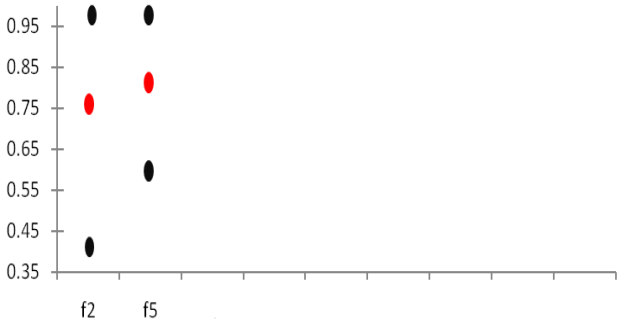
| Feature | f1 | f2 | f3 | f4 | f5 | f6 | f7 | f8 |
|---|---|---|---|---|---|---|---|---|
| Pos. score | 0.9651 | 0.9777 | 0.9685 | 0.9787 | 0.9881 | 0.9896 | 0.9863 | 0.9917 |
| Neg. score | 0.3131 | 0.4126 | 0.4004 | 0.2674 | 0.34 | 0.3811 | 0.2124 | 0.3017 |
| Sat. score | 0.7043 | 0.7516 | 0.7413 | 0.6942 | 0.7289 | 0.7462 | 0.6768 | 0.7157 |




Feature $f_2$ has agrmax of satisfied score, it is moved from F to $F_C$: F = $\{f_1, f_3, \ldots, f_8\}$, $F_C$ = $\{f_2\}$

***Running step 2***: calculate satisfied score for all left features, combination with $f_2$, by using evaluation function $g(f_2 \wedge f_i)$, with $i$ = {1,..,8}\{2}

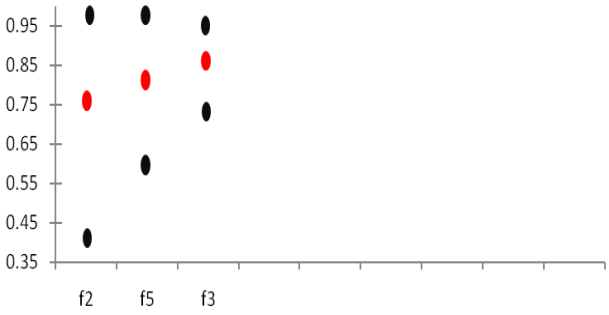
| Feature | f2∧f1 | f2 | f2∧f3 | f2∧f4 | f2∧f5 | f2∧f6 | f2∧f7 | f2∧f8 |
|---|---|---|---|---|---|---|---|---|
| Pos. score | 0.9463 | | 0.9525 | 0.9485 | 0.9696 | 0.9586 | 0.9509 | 0.9843 |
| Neg. score | 0.4224 | | 0.5916 | 0.4304 | 0.5929 | 0.4042 | 0.3343 | 0.4026 |
| Sat. score | 0.7368 | | 0.8082 | 0.7413 | 0.8189 | 0.7369 | 0.7043 | 0.7516 |




Combination of features ($f_2 \wedge f_5$) has agrmax of satisfied score, so $f_5$ is moved from F to $F_C$:
F = {$f_1$, $f_3$, $f_4$, $f_6$, $f_7$, $f_8$}; $F_C$ = {$f_2$, $f_5$}

***Running step 3***: calculate satisfied score for all left features, combination with ($f_2 \wedge f_5$), by using evaluation function $g(f_2 \wedge f_5 \wedge f_i)$, with $i$ = {1,..,8}\{2,5}

| Feature | f25∧f1 | f2 | f25∧f3 | f25∧f4 | f5 | f25∧f6 | f25∧f7 | f25∧f8 |
|---|---|---|---|---|---|---|---|---|
| Pos. score | 0.9312 | | 0.9485 | 0.9136 | | 0.9215 | 0.9359 | 0.9641 |
| Neg. score | 0.5418 | | 0.7285 | 0.4412 | | 0.4297 | 0.4028 | 0.4258 |
| Sat. score | 0.7754 | | 0.8605 | 0.7247 | | 0.7248 | 0.7227 | 0.7488 |

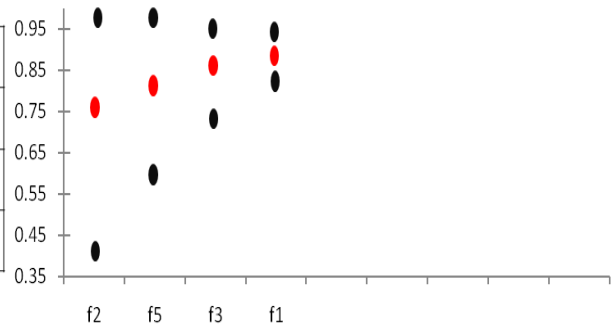



Combination of features ($f_2 \wedge f_5 \wedge f_3$) has agrmax of satisfied score, so $f_5$ is moved from F to $F_C$: F = {$f_1$, $f_4$, $f_6$, $f_7$, $f_8$}; $F_C$ = {$f_2$, $f_5$, $f_3$}

***Running step 3***: calculate satisfied score for all left features, combination with ($f_2 \wedge f_5 \wedge f_3$), by using evaluation function $g(f_2 \wedge f_5 \wedge f_3 \wedge f_i)$, with $i$ = {1,..,8}\{2,5,3}

| Feature | f253∧f1 | f2 | f3 | f253∧f4 | f5 | f253∧f6 | f253∧f7 | f253∧f8 |
|---|---|---|---|---|---|---|---|---|
| Pos. score | 0.934 | | | 0.9084 | | 0.9054 | 0.9023 | 0.944 |
| Neg. score | 0.8259 | | | 0.7706 | | 0.5631 | 0.7724 | 0.7421 |
| Sat. score | 0.8907 | | | 0.8533 | | 0.7685 | 0.8504 | 0.8632 |

agrmax = 0.8907

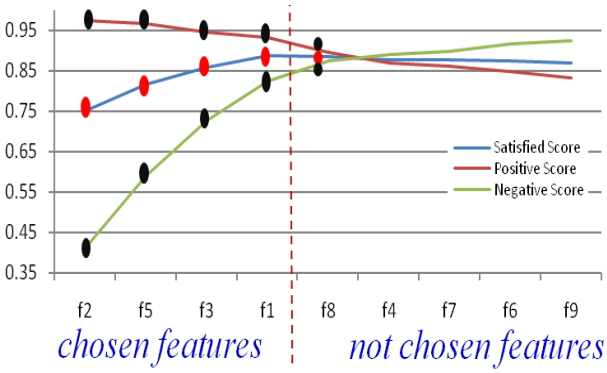



Combination of features ($f_2 \wedge f_5 \wedge f_3 \wedge f_1$) has agrmax of satisfied score, so $f_1$ is moved from F to $F_C$: F = { $f_4$, $f_6$, $f_7$, $f_8$}; $F_C$ = {$f_2$, $f_5$, $f_3$, $f_1$}

***Running step 4***: calculate satisfied score for all left features, combination with ($f_2 \wedge f_5 \wedge f_3 \wedge f_1$), by using evaluation function $g(f_2 \wedge f_5 \wedge f_3 \wedge f_1 \wedge f_i)$, with $i$ = {1,..,8}\{2,5,3,1}

| Feature | f1 | f2 | f3 | f2531∧f4 | f5 | f2531∧f6 | f2531∧f7 | f2531∧f8 |
|---|---|---|---|---|---|---|---|---|
| Pos. score | | | | 0.884 | | 0.8927 | 0.8952 | 0.897 |
| Neg. score | | | | 0.8353 | | 0.8019 | 0.7766 | 0.8754 |
| Sat. score | | | | 0.8645 | | 0.8564 | 0.8477 | **0.8883** |

$agrmax_{present}(0.8883) < argmax_{previous}(0.8907)$




Combination of features ($f_2 \wedge f_5 \wedge f_3 \wedge f_1$) has agrmax of satisfied score. Howerver, at this step, the stop conditional is satisfied: $agrmax_{present} = 0.8883 < agrmax_{previous} = 0.8907$. Then, the feature $f_8$ is not moved from F to $F_C$, and our algorithm stops here. Finally, four features that are good for recognizing object are $F_C = \{f_2, f_5, f_3, f_1\}$.

## 5.4. Experimental results of automatically choosing features

To evaluate the proposed approach, we performed two experiences. Firstly, we run the proposed automatic feature selection algorithm individually to get list of good features for every kinds of object. Secondly, these good features will be put into our detecting system (as described in chapter 4) for recognizing object to compare detection result between automatic with manual feature selection. This section presents the first evaluation, while the second evaluation is showed in section 5.5.

The proposed approach was tested on ten categories of object including eight objects as in chapter 4 (front/rear view car, side view car, bicycle, motorbike, train, plane, horse and sheep) and two new classes (tower and flower object). With eight old class, the same training image database as in chapter 4 is used in this evaluation. Image database of tower is collected manually from internet, while flower object database is mainly based on Oxford[7] flower database, and partly saved from web.

We test with seven features to describe images: 'edge', 'corner', 'HoG', 'circle', 'line', 'SURF', and 'color'. Bias $\omega_p$ and $\omega_n$ in the algorithm are set to 0.6 and 0.4 respectively.

[7] http://www.robots.ox.ac.uk/~vgg/data/flowers/

| Object | Edge | Corner | Line | Circle | HoG | SURF | Color |
|---|---|---|---|---|---|---|---|
| F/r car | 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| Side car | 1 | 1 | 0 | 1 | 1 | 1 | 0 |
| Bike | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| Train | 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| Aero plane | 1 | 1 | 1 | 0 | 0 | 1 | 0 |
| Motorbike | 1 | 1 | 0 | 0 | 0 | 1 | 1 |
| Horse | 1 | 1 | 1 | 0 | 0 | 1 | 1 |
| Sheep | 1 | 1 | 0 | 0 | 0 | 1 | 1 |
| Tower | 1 | 1 | 1 | 0 | 1 | 0 | 0 |
| Flower | 1 | 1 | 0 | 0 | 1 | 1 | 0 |

**Table 5.1 Object with automatically chosen features**

Value of positive bias is higher then negative one, because, the goal of our system is to detect instant of object in an image, not to say that there is no object in that image. Result of automatically choosing features is in table 3, where zero (0) means that feature is not good for object, and one (1) represents we should use this feature to detect object. The outcome of this algorithm is predictable, it is almost fit with what we have done by hand. Only one thing is not as we though is that color feature is useful for motorbike object (value 1 at the cell of column Color and row Motorbike). After checking the motorbike database, we recognize that most of motorbike-images from CALTECH 256 have white background and their colors only vary in green, brown, red. That is the reason why with this database, color is helpful for detecting motorbike object.

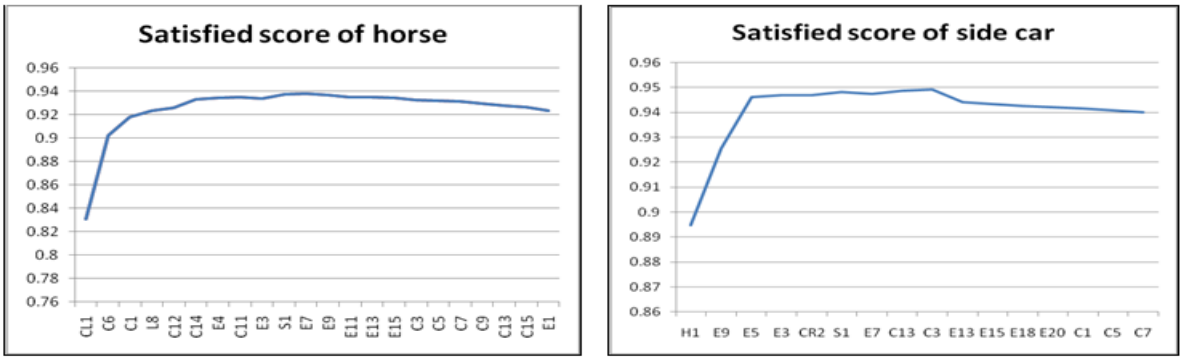



**Figure 5.2 Satisfied scored for automatically choosing feature**

In the fig. 11, fig. 12, and fig. 13 detail of automatically choosing feature for horse object and side view car object is described. From set of features, we choose one feature with the

highest satisfied score (calculated as in section 4). We combine every feature with the first chosen feature, and then we recalculate the satisfied score. We will pick the feature which makes the satisfied score maximum. This feature then is moved to chosen feature set. Next running time, both two chosen features are integrated with new one to compute new satisfied score. By doing this way, the satisfied score increases slowly until it rises to a superior value, which is the maximum value. After that, if we combine more features, result will be worse. If the score is lower than previous in three running times, we will stop. Our algorithm will reverse to the peak, and only these features before gaining the top will be chosen for that object. For example, with horse object these are chosen-features: CL (color), C (corner), L (line), E (edge) and S (SURF). Other while, with side-view car, H (HoG) is the best feature following by E (edge), CR (circle), E (edge) and C (corner).

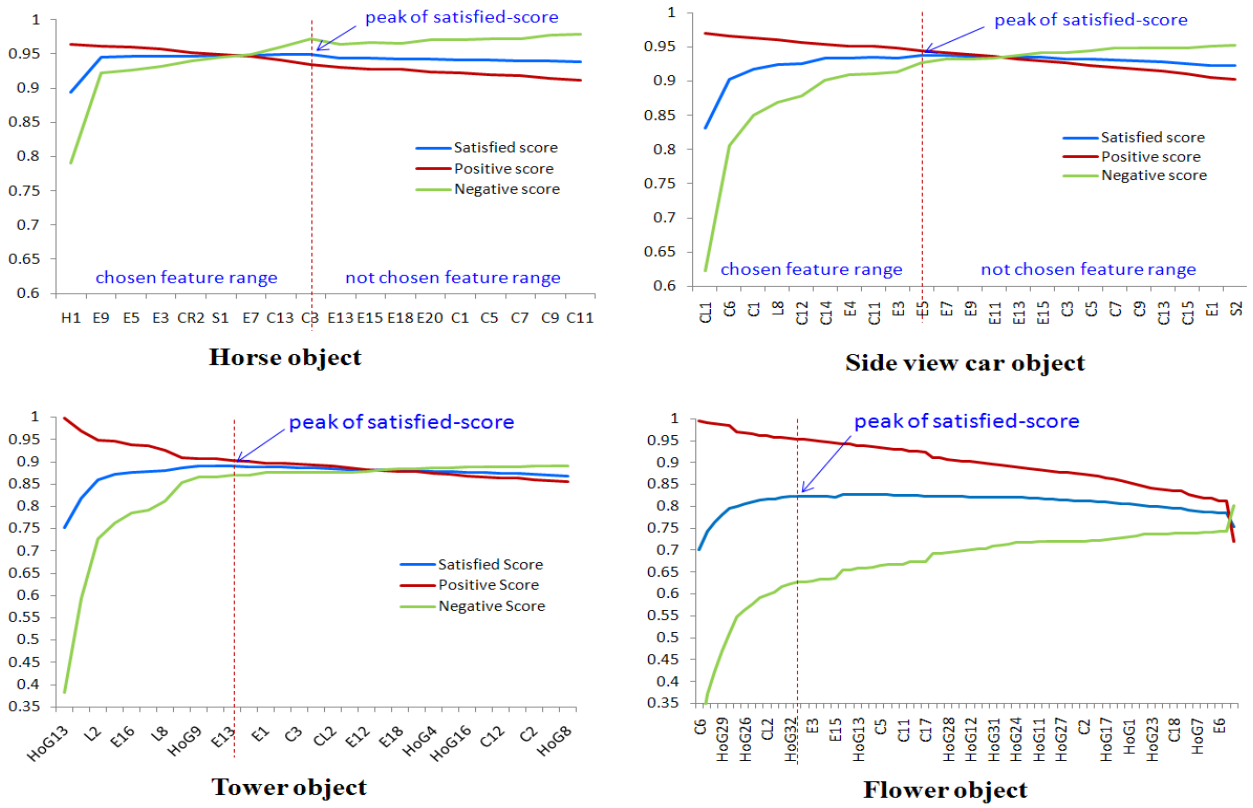



**Figure 5.3 Satisfied, positive and negative score of some objects**

With the first feature, generally *positive score* $pos(f_i,F_C) = |sat(F_C \cup \{f_i\}|T_{pos})|*1/|T_{pos}|$ is very large because most of image in $T_{pos}$ satisfy this feature. Other while, the *negative score* $neg(f_i,F_C) = (|T_{neg}|- |sat((F_C \cup \{f_i\})|T_{neg})|)*1/|T_{neg}|$ is small. When the more features are added to

chosen feature set, the smaller positive score is, and the larger negative score becomes. Totally, the *satisfied score* $g(f_i,F_C) = \omega_p*pos(f_i,F_C) + \omega_n*neg(f_i,F_C)$ (with $\omega_p = 0.6$ and $\omega_n = 0.4$), get larger until it reach the prime value, represented by red line. After this peak, if more features are integrated into $F_C$, the satisfied score gets lower. After the satisfied score becomes worse k times (in our evaluation, k = 3), the system stops and reverses to get the prominent value (at red interrupted-line). Left side of this red line is chosen features and not good features are on the right side.

## 5.5. Object detection by automatic feature selection

To evaluate the performance of the proposed approach, we have performed a number of experiments and comparisons to demonstrate the importance of feature selection for object detection. First, the proposal algorithm for automatic feature selection is run to get list of good features for every object. List of good features for a kind of object is showed in table 5.1. Then, these good features corresponding to a specific object would be put in to our detecting system (as described in chapter 4) to detect object.

### *5.3.1. Result of object detection with auto-choosen features*

This part present results of applying good features for detecting object. We ran several experiments with ten categories of object. Table 5.2 shows average precision (AP) and average recall (AR) of training and testing stage.

| Object | TRAINING STAGE | | TESTING STAGE | |
|---|---|---|---|---|
| | Average Precision | Average Recall | Average Precision | Average Recall |
| F/r car | 96.48% | 90.21% | 95.12% | 90.14% |
| Side car | 97.20% | 94.92% | 94.21% | 91.73% |
| Bike | 85.80% | 81.32% | 84.02% | 79.14% |
| Train | 87.24% | 77.16% | 83.65% | 75.31% |
| Aero plane | 86.65% | 84.55% | 85.75% | 84.51% |
| Motorbike | 90.23% | 87.32% | 89.38% | 85.47% |
| Horse | 88.93% | 82.09% | 87.81% | 75.40% |
| Sheep | 87.32% | 75.91% | 86.25% | 73.24% |
| Tower | 89.07% | 90.39% | 84.33% | 82.64% |
| Flower | 82.71% | 75.15% | 81.57% | 74.42% |

**Table 5.2 AP & AR at training/testing stage of automatically choosing features**


**Figure 5.4 Examples of flower detection using auto good feature selection**

Some examples of using good features from automatic selection for detecting flower object are showed in figure 5.4. In this figure, some concrete examples of correct and incorrect detections are provided. False positives commonly arise in highly cluttered areas, but sometimes they are due to inherent properties of the detection model. The first image in the third row is a rare example of a failure due to its shape is seldom seen in nature.

### *5.3.2. Comparasion between auto- and manual- selection*

Beside the total result for all ten categories is showed in table 5.2, we also make a comparison between automatic and manual feature selection. This comparison evaluates the accuracy of proposal algorithm. Table 5.3 visualizes the equivalence between these two methods by the detection accuracy of individual object. Features selected by the proposal algorithm prove to be significantly more appropriate for the experimental object detection tasks than features manually selected. Despite their apparent simplicity, these features collectively possess sufficient strength to bring about relatively high detection accuracy. Our results for the ten objects dataset

describes that the proposal algorithm for automatic feature selection gets a high belief for detecting new object.

| Object | MANUAL | | AUTOMATIC | |
|---|---|---|---|---|
| | Average Precision | Average Recall | Average Precision | Average Recall |
| F/r car | 97.40% | 89.71% | 95.12% | 90.14% |
| Side car | 95.83% | 92.38% | 94.21% | 91.73% |
| Bike | 83.82% | 78.46% | 84.02% | 79.14% |
| Train | 84.05% | 74.10% | 83.65% | 75.31% |
| Aero plane | 85.59% | 87.56% | 85.75% | 84.51% |
| Motorbike | 95.63% | 85.35% | 89.38% | 85.47% |
| Horse | 88.37% | 70.72% | 87.81% | 75.40% |
| Sheep | 86.13% | 71.93% | 86.25% | 73.24% |
| Tower | | | 84.33% | 82.64% |
| Flower | | | 81.57% | 74.42% |

**Table 5.3 Comparison between manual and automatic feature selection**

Results of our detecting system using auto chosen features are also compared with result of PASCAL 2010. Table 5.4 presents this equivalence.

| Object | Our system | PASCAL VOC 2010[8] | |
|---|---|---|---|
| | Average Precision | Average Precision | Authors |
| Front/rear car | 95.12% | 49.10% | **UOCTTI_LSVM_MDPM** |
| Side car | 94.21% | | |
| Bicycle | 84.02% | 55.30% | **NLPR_HOGLBP_MC_LCEGCHLC** |
| Train | 83.65% | 50.30% | **MITUCLA_HIERARCHY** |
| Aero plane | 85.75% | 58.40% | **UVA_GROUPLOC** |
| Motorbike | 89.38% | 56.30% | **NUS_HOGLBP_CTX_CLS_RESCORE_V2** |
| Horse | 87.81% | 51.90% | **NUS_HOGLBP_CTX_CLS_RESCORE_V2** |
| Sheep | 86.25% | 37.80% | **UVA_DETMONKEY** |
| Tower | 84.33% | 84.00% | **HENA_LU_DU** [9] |
| Flower | 81.57% | 80.00% | **JZU_HONG_CHEN_LI**[10] |

**Table 5.4 Comparison between our system with PASCAL 2010**

[8] *http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2010/results/ (about 20 categories)*

[9] *Lu Yang et. el., 2nd Intl Asia Conf on Informatics in Control, Automation and Robotics 2010, p.p 349-352*

[10] *Hong et al. / J Zhejiang Univ SCI 2004 5(7):764-772 http://www.zju.edu.cn/jzus*


**Figure 5.5 PASCAL 2010 contest with 20 categories**

## 5.6. Summary

This chapter provides a comprehensive detail of our proposal algorithm for automatic feature selection for detecting object. We introduce this algorithm based on probability method. This algorithm includes three main steps: initialize step with sequential forward selection (SFS), evaluation step using fitness function *g(F)* to calculate satisfied score, and ending step with two stopping conditions. This new feature selection algorithm can be incorporated into detecting system in previous chapter. This algorithm is generic and easily adaptable to new objects with little re-programming. More important is that we can add many features to this algorithm as possible, without any modification. We evaluate the performance of automatically choosing semantic attribute features on the large data set as used in chapter 4. The outcome of this algorithm is predictable. Moreover, based on the chosen features, we apply them for detecting ten objects and the accuracy of detection result is very satisfactory. Due to this, the effectiveness of the developed selection strategy was experimentally confirmed. Whereas the selection algorithm undoubtedly plays a significant role, the overall success of the method should also be attributed to the feature extraction and object detection modules. Last but not least, our algorithm doesn't need to confirm number of selected features as other approach, this number would be auto computed when our algorithm finish running.

# Chapter 6 - Conclusion and Future work

This thesis has introduced a general approach and techniques for object detecting by automatic feature selection, which is a subject that is receiving rapidly increasing attentions in recent years. One of the most important aims of this research is to bridge the semantic gap, which is considered as a vital problem existing in the traditional content based image retrieval systems. In this final chapter, we will draw together and summarize the main conclusions of the research undertaken by the author, and give some pointers to future research following the work presented in the earlier chapters.

## 6.1. Conclusion

Computer vision techniques that build up low-level image features for describing and comparing image content are the foundation of research in object detection. The choice of an appropriate feature mechanism has a large influence on the performance of an object detection system. In chapter 3, we have introduced a number of such techniques. Feature extraction is the main step of image description. Depend on the interesting region chosen for extracting feature, it can be global feature (if whole image is chosen) or local feature (or part-based feature corresponding to a part of image). Both global and local feature are used for description image. Global feature describes the context information of that image, while local feature presents more detail about some small regions. Due to that, combination both of them often gets better result. There are a great number of different feature descriptors, from basic edge, corner, circle, line, color, descriptors to more advanced descriptors such as HoG, SURF. In some cases, the quantization step is adopted to group feature descriptors into classes, so that CBIR or object detection systems can avoid dealing with a massive number of feature descriptors which are possibly of high dimension.

Chapter 4 proposed a novel approach for improving performance of object detection. Taking into account that both local and global features may provide useful information to recognize object, the combination of multi features to construct a system gets a good performance. We have presented a general feature-based object detection approach with combination of many features such as edge, corner, HoG, SURF and color. This method selects features, and learns a hierarchical classifier by combining cascade classifiers to detect objects in

images. Extensive object detection experiments show high detection rates with relatively low numbers of false detections. These results illustrate the high discriminate power of combination both local and global features and the effectiveness and robustness of the hierarchical object detection approach. The proposed approach is able to detect objects that consist of distinguishable parts in spatial configurations not only natural objects such as horse, sheep; but also man-made objects such as side-view car, front/rear view car, bicycle, motorbike, train and aeroplane. In summary, the results show that the object representation using combination multi features is general to different kinds of object classes, and our feature selection methods are efficient to extract informative class specific features for object detection. We evaluate a large image database and the outcome is satisfactory.

Finally, in chapter 5, we took a step forward to automatically choosing features. In this chapter, we proposed a novel model for automatic feature selection. We introduce this algorithm based on probability method. This algorithm includes three main steps: initialize step with sequential forward selection (SFS), evaluation step using fitness function *g(F)* to calculate satisfied score, and ending step with two stopping conditions. This new feature selection algorithm can be incorporated into detecting system in previous chapter. This algorithm is generic and easily adaptable to new objects with little re-programming. More important is that we can add many features to this algorithm as possible, without any modification. We evaluate the performance of automatically choosing semantic attribute features on the large data set as used in chapter 4. The outcome of this algorithm is predictable. Moreover, based on the chosen features, we apply them for detecting ten objects and the accuracy of detection result is very satisfactory. Due to this, the effectiveness of the developed selection strategy was experimentally confirmed. Whereas the selection algorithm undoubtedly plays a significant role, the overall success of the method should also be attributed to the feature extraction and object detection modules. Last but not least, our algorithm doesn’t need to confirm number of selected features as other approach, this number would be auto computed when our algorithm finish running. Experimental results confirmed that automatic feature selection performance is enhanced. It also implies that the accuracy of choosing good features for an object is very important to an object detecting system. Improving the process of feature selection can lead to improvement in system performance.

In summary, this thesis has made a number of contributions to the community of automatic feature selection as well as object detection. A list of the contributions has been outlined in the introduction. In the following, we reaffirm the novelties associated with these contributions.

- An in depth investigation into some of the quality issues with image data-sets that are used for research on object detecting from static image.
- The demonstration of the potentials of many feature-extract-algorithms such as Haris Coner detector, Hough Transform, SURF-64/128, ...
- The development of a model for descripting image throuhg feature vector space that improves runing-time of object detection.
- The development of an robust approach to finding object classes in an unlabelled still image collection using both local and global features: edge, corner, HoG, circle, line, color, SURF.
- The development of a newest algorithm (named best-fixed algorithm) for un-manually selecting best features to detect a specific object.
- The implement of the proposed best-fixed algorithm on a large image database with many object categories from natural objects (flower, animal, horse, sheep ...) to man-made ones (car, bike, tower, plane...)
- The evaluation of proposed best-fixed algorithm's reslut, comparation with outcome of manually choosing feature.

## 6.2. Future work

Since the emerging of the technique of object detection almost many decades ago, researchers from both computer vision and machine learning societies have devoted a lot of efforts to advancing it. Although very promising results have been achieved so far, research in this field still has a long way to go before this technique can be utilized in our daily life. Indeed, object detection is a very challenging problem that needs the collaboration of different techniques from a variety of disciplines in order for it to achieve success. This research develops a novel and general approach for object detection. Although in this method, we combine both local and global features, but until this time, only seven features are used in this research: edge, corner, HoG, cirle, line, SURF and color. There are many other robust features needed to add to

our system such as texture, shape, … The more features we use, the better performance of our system is. This is one of our future tasks to improve detection result.

This research has also solved the automatic feature selection problem. Under the proposed framework, different with other state-of-art image retrieve systems, we also set up an algorithm for automatically choosing feature. This algorithm does not only determine which feature is good for which object, but it also unintentional compute the threshold value for each feature. Up to present, this is the first proposed algorithm that can do both of these tasks. But in this research, we only use *"and"* operator $(f_i \wedge f_{i+1} \wedge \ldots \wedge f_{i+k})$ for combining features. What does happen if both *"and"* and *"or"* operator $(((f_i \wedge f_{i+1}) \vee f_{i+2}) \wedge \ldots \wedge (f_{i+k-1} \vee f_{i+k}))$ take part in detecting? With $n$ features and only two fundamental operators $(\wedge, \vee)$, there are about $2^{n-1}$ combinations. It is really a huge number, and we cannot check all the cases due to large running time. We should find a suitable way to combine with both two basic operators, but in a limited running time. This is truly a great challenge for automatically choosing feature. If we can solve this problem, the accuracy of our method will become more believable.


**Figure 6.1 Example of small object cannot be detected**

Applying auto chosen features for detecting object has also done in this thesis. Experimental results on ten categories from man made objects to natural objects also get good result. But, in some special case, if the object in the input image is too small compared with the whole image, our system cannot recognize. Figure 6.1 shows some examples. How to determine the best scale for detecting is also a challenge and interesting future work.

# References


[1] N. Snavely, S.M. Seitz, R. Szeliski, "Photo tourism: exploring photo collections in 3D," *SIGGRAPH*, 2006.

[2] T. Yeh, J.J. Lee, T. Darrell, "Photo-based question answering," *ACM Multimedia*, 2008.

[3] Y. Li, K.W. Wan, X. Yan, C. Xu, "Real time advertisement insertion in baseball video based on advertisement effect," *ACM Multimedia*, 2005.

[4] W.-S. Liao, K.-T. Chen, W.H. Hsu, "Adimage: video advertising by image matching and ad scheduling optimization," *ACM SIGIR*, 2008.

[5] X.-J. Wang, L. Zhang, F. Jing, W.-Y. Ma, "Annosearch: Image auto-annotation by search," *CVPR*, 2006.

[6] J. Hays, A.A. Efros, "Im2gps: estimating geographic information from a single image," *CVPR*, 2008.

[7] H.A. Rowley, S. Baluja, T. Kanade, "Neural network-based face detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20 (1) (1998) 29–38.

[8] C. Garcia, M. Delakis, "Convolutional face finder: a neural architecture for fast and robust face detection," *IEEE Tran on Pattern Analysis and Machine Intelligence 26* (11) (2004).

[9] E. Osuna, R. Freund, F. Girosi, "Training support vector machines: an application to face detection," *Proceedings of the 1997 IEEE CVPR*, 1997, pp. 130–136.

[10] C.P. papageprgiou,T. Poggio, "A training object system: car detection in static images," *MIT AI Memo No. 180*, 1999.

[11] H. Schneiderman, T. Kanade, "A statistical method for 3D object detection applied to faces and cars," *Proc of the 2000 IEEE Conference on CVPR* (2000) 746–751.

[12] P. Viola, M. Jones, "Rapid object detection using a boosted cascade of simple features," *Proc of the 2001 IEEE Conference on CVPR (2001)* 511–518.

[13] S.Z. Li, L. Zhu, Z.Q. Zhang, A. Blake, H.J. Zhang, H. Shum, "Statistical learning of multi-view face detection," *Proc of the 7th European Conference on Computer Vision 4* (2002).

[14] X.R. Chen, A. Yuille, "Detecting and reading text in natural scenes," *Proc of the IEEE International Conference on CVPR* (2004) 366–373.

[15] K.K. Sung, T. Poggio, "Example-based learning for view-based human face detection", *IEEE Transactions on Pattern Analysis and Machine Intelligence 20 (1)* (1998) 39–50.

[16] C.J. Liu, "A Bayesian discriminating features method for face detection", *IEEE Transactions on Pattern Analysis and Machine Intelligence 25*(2003) 725–740.

[17] B. Menser, F. Muller, "Face detection in color images using principal component analysis," *Proceedings of the Seventh International Congress on Image Processing and its Applications,* 1999, pp. 13–15.

[18] M.H. Yang, N. Ahuja, D. Kriegman, "Face detection using mixtures of linear subspaces," *the 4th IEEE Intl Conference on Automatic Face and Gesture Recognition,* 2000, pp. 70–76.

[19] A. Mohan, C. Papageorgiou, T. Poggio, "Example-based object detection in images by components," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 2001.

[20] M.V. Naquest, S. Ullman, "Object recognition with informative features and linear classification," *the 9th International Conference on Computer Vision,* 2003, pp. 281–288.

[21] S. Agarwal, A. Awan, D. Roth, "Learning to detect objects in images via a sparse, part-based representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence 26* (11) (2004) 1475–1490.

[22] R. Fergus, P. Perona, A. Zisserman, "Object class recognition by unsurpervise scale-invariant learning," *the 9th International Conference on CVPR 2* (2003) 264–271.

[23] B. Leibe, A. Leonardis, B. Schiele, "Combined object categorization and segmentation with an implicit shape model," *ECCV2004 Workshop on Statistical Learning in Computer Vision,* 2004.

[24] B. Leibe, B. Schiele, "Scale-invariant object categorization using a scaleadaptive mean-shift search," *Proceedings of the DAGM'04 Annual Pattern Recognition Symposium, 3175, Springer LNCS, Berlin,* 2004, pp. 145–153.

[25] Y.Y. Lin, T.L. Liu, "Robust face detection with multi-class boosting," *Proceedings of the IEEE Intl Conf on Computer Vision and Pattern Recognition* 1 (2005) 680–687.

[26] Y. Amit, D. Geman, X.D. Fan, "A coarse-to-fine strategy for multiclass shape detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence 26 (12)* (2004).

[27] F.F. Li, R. Fergus, P. Perona, "A bayesian approach to unsupervised oneshot learning of object categories," *the Ninth International Conference on Computer Vision,* 2003.

[28] F.F. Li, R. Fergus, P. Perona, "Learning generative visual models from few training examples: an incremental Bayesian approach tested on 101 object categories," *Conference on CVPR Workshop on Generative-Model Based Vision,* 2004.

[29] A.Torralba, K.P. Murphy, W.T. Freeman, "Sharing features: efficient boosting procedures for multiclass object detection," *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR2004),* 2004, pp. 762–769.

[30] Z.W. Tu, X.R. Chen, A.L. Yuille, S.C. Zhu, "Image parsing: unifying segmentation, detection and recognition," *International Journal of Computer Vision 63* (2005) 113–140.

[31] B.Schiele, "Object recognition using multidimensional receptive field histograms," *PhD Thesis, I.N.P. Grenoble,* English translation, 1997.

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

[33] K.I. Kim, K. Jung, J.H. Kim, "Texture-based approach for text detection in images using support vector machines and continuously adaptive mean shift algorithm," *IEEE Transactions on Pattern Analysis and Machine Intelligence 25 (12)* (2003) 1631–1639.

[34] F. Bernhard, K. Christian, "Real-time face detection using edge-orientation matching," *the Third Intl Conference Audio- and Video-Based Biometric Person Authentication,* 2001.

[35] C. Garcia, G. Tziritas, "Face detection using quantized skin color regions merging and wavelet packet analysis," *IEEE Transactions on Multimedia 1 (3)* (1999) 264–277.

[36] M. Bicego et al., "On the use of SIFT features for face authentication," in *Proc. of the Int. Conf. on CVPR*, June 2006.

[37] Mohamed Rizon et al., "Object detection using Circular Hough Transform," *American Journal of Applied Sciences 2*, p.p. 1606-1609, 2005.

[38] H. Harzallah, F. Jurie, C. Schmid, "Combining efficient object localization and image classification," *ICCV09*, Kyoto, Japan, 2009.

[39] H. Bay, T. Tuytelaars, and L. V. Gool, "Surf: Speeded up robust features," *ECCV06*, Graz, Austria, May 2006.

[40] Zhenfeng Zhu., J. Uchimura, "Car detection based on multi-cues integration," *ICPR04*, Cambridge, UK, Aug. 2004.

[41] J. Zhang, M. Marszałek, S. Lazebnik, and C. Schmid, "Local features and kernels for classification of texture and object categories: a comprehensive study," *IJCV07*, 73(2), 2007.

[42] L. Fei-Fei, R. Fergus, and P. Perona, "Learning generative visual models from few training examples: An incremental Bayesian approach tested on 101 object categories," *Computer Vision and Image Understanding*, 106(1), 2007.

[43] A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain, "Content-based image retrieval at the end of the early years," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 12, pp. 1349-1380, 2000.

[44] V. Ferrari and A. Zisserman, "Learning visual attributes," *Advances in Neural Information Processing Systems*, 2008.

[45] Y. Jing, S. Baluja, and H. Rowley, "Canonical image selection from the web," *the 6th ACM Itnl Conference on Image and Video retrieval*, New York, USA, 2007, pp. 280-287.

[46] J. Uijlings, A. Smeulders, and R. Scha, "What is the spatial extent of an object?," *IEEE CVPR*, 2009.

[47] T. Arni, P. Clough, M. Sanderson, and M. Grubinger, "Overview of the ImageCLEFphoto 2008 photographic retrieval task," *Working Notes for the CLEF 2008 Workshop*, 2008.

[48] Marcin Marszałek, Cordelia Schmid, Hedi Harzallah, and Joost van de Weijer, "Learning object representations for visual object class recognition," *ICCV07*, October 2007.

[49] J. van deWeijer, C. Schmid, and J. Verbeek, "Learning color names from real-world images," *IEEE CV PR*, 2007.

[50] J. Vogel and B. Schiele, "Natural scene retrieval based on a semantic modeling step," *Intl. Image and Video Retrieval*, 2004.

[51] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth, "Describing objects by their attributes," *IEEE CV PR*, 2009.

[52] C. Lampert, H. Nickisch, and S. Harmeling, "Learning to detect unseen object classes by between-class attribute transfer," *IEEE Intl CV PR*, 2009.

[53] Sonya Coleman, Bryan Scotney, and Dermot Kerr, "Integrated Edge and Corner Detection," *Intl Conference on Image Analysis and Processing*, Modena, Italy, Sep 2007.

[54] D. G. Lowe, "Distinctive image features from scale invariant keypoints," *IJ CV, 60(2)*, pages 91–110, 2004.

[55] David Monzo, Alberto Albiol, Jorge Sastre, Antonio Albiol, "HOG-EBGM VS. GABOR-EBGM," *IEEE Intl Conference on Image Processing*, San Diego, CA, Oct. 2008.

[56] Navneet Dalal and Bill Triggs, "Histograms of Oriented Gradients for Human Detection," *CVPR*, CA, USA, June 2005.

[57] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," *IEEE CVPR*, 2006.

[58] Gabriella Csurka, Christopher R. Dance, Lixin Fan, Jutta Willamowski, and Cédric Bray, *"Visual Categorization with Bags of Keypoints*," Xerox Research Centre Europe, 2004.

[59] S. Di Zenzo, "Note: A note on the gradient of a multi-image," *Computer Vision, Graphics, and Image Processing*, vol. 33, no. 1, pp. 116–125, 1986.

[60] J. Bigun, "Pattern recognition in images by symmetry and coordinate transformations," *Computer Vision and Image Understanding*, vol. 68, no. 3, pp. 290–307, 1997.

[61] J. Bigun, G. Granlund, and J. Wiklund, "Multidimensional orientation estimation with applications to texture analysis and opitcal flow," *IEEE trans. on pattern analysis and machine intelligence*, vol. 13, no. 8, pp. 775–790, 1991.

[62] O. Hansen and J. Bigun, "Local symmetry modeling in multi-dimensional images," *pattern Recognition Letters*, vol. 13, pp. 253–262, 1992.

[63] J. van de Weijer, L. van Vliet, P. Verbeek, and M. van Ginkel, "Curvature estimation in oriented patterns using curvilinear models applied to gradient vector fields," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 23, no. 9, pp. 1035–1042, 2001.

[64] S. Shafer, "Using color to seperate reflection components," *COLOR research and application*, vol. 10, pp. 210–218, Winter, 1985.

[65] T. Gevers and H. Stokman, "Robust histogram construction from color invariants for object recognition," *IEEE Trans. On Pattern Analysis and Machine Intelligence (PAMI)*, vol. 26, no. 1, pp. 113–118, 2004.

[66] T. Gevers and A.W. M. Smeulders, "Color based object recognition," *Pattern Recognition*, vol. 32, pp. 453–464, March 1999.

[67] G. Klinker and S. Shafer, "A physical approach to color image understanding," *Int. Journal of Computer Vision*, vol. 4, pp. 7–38, 1990.

[68] J. Geusebroek, R. van den Boomgaard, A. Smeulders, and H. Geerts, "Color invariance," *IEEE Trans. Pattern Analysis Machine Intell.*, vol. 23, no. 12, pp. 1338–1350, 2001.

[69] J. van de Weijer, T. Gevers, and J. Geusebroek, "Edge and corner detection by photometric quasi-invariants," *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2005.

[70] D. Brown, I. Craw, J. Lewthwaite, "A SOM based approach to skin detection with application in real time systems," *BMVC01*, 2001.

[71] C. Garcia, G. Tziritas, "Face detection using quantized skin color regions merging and wavelet packet analysis," *IEEE Trans. Multimedia 1 (3)* (1999) 264–277.

[72] S. McKenna, S. Gong, Y. Raja, "Modeling facial colour and identity with Gaussian mixtures," *Pattern Recognition 31 (12) (1998)* 1883–1892.

[73] D. Saxe, R. Foulds, "Toward robust skin identification in video images," *AFGR96*, 1996.

[74] K. Sobottka, I. Pitas, "A novel method for automatic face segmentation, facial feature extraction and tracking," *Signal Process. Image Commun. 12* (1998) 263–281.

[75] Q.H. Thu, M. Meguro, M. Kaneko, "Skin-color extraction in images with complex background and varying illumination," *Sixth IEEE Workshop on Applications of Computer Vision*, 2002.

[76] Y. Wang, B. Yuan, "A novel approach for human face detection from color images under complex background," *Pattern Recognition 34 (10)* (2001) 1983–1992.

[77] Q. Zhu, K.-T. Cheng, C.-T. Wu, Y.-L. Wu, "Adaptive learning of an accurate skin-color model," *AFGR04*, 2004.

[78] B.D. Zarit, J.B. Super, F.K.H. Quek, "Comparison of five color models in skin pixel classification," *ICCV99*, 1999.

[79] J.C. Terillon, M. David, S. Akamatsu, "Detection of human faces in complex scene images by use of a skin color model and of invariant Fourier–Mellin moments," *ICPR98*, 1998, pp. 350–1355.

[80] Paul Viola and Michael Jones, "Rapid object detection using a boosted cascade of simple features", *CVPR01*, 1:511, 2001.

[81] Z. Sun, G. Bebis, R. Miller, "Boosting object detection using feature selection", *IEEE Intl Conference on Advanced Video and Signal Based Surveillance*, July 2003, pp. 290–296.

[82] P. Viola, M. Jones, "Rapid object detection using a boosted cascacd of simple features," *Proceedings on Computer Vision and Pattern Recognition*, 2001.

[83] Z. Sun, X. Yuan, G. Bebis, S. Louis, "Neural-network-based gender classi1cation using genetic eigen-feature extraction," *IEEE Intl Joint Conf on Neural Networks*, May 2002.

[84] Z. Sun, G. Bebis, X. Yuan, S. Louis, "Genetic feature subset selection for gender classification: a comparison study," *IEEE Workshop on Applications of Computer Vision*, December 2002.

[85] Z. Sun, G. Bebis, R. Miller, "Evolutionary gabor filter optimization with application to vehicle detection," *IEEE International Conference on Data Mining*, November 2003.

[86] A. Gyaourova, G. Bebis, I. Pavlidis, "Infrared and visible image fusion for face recognition," *European Conference on Computer Vision*, May 2004.

[87] B. Bhanu, Y. Lin, "Genetic algorithm based feature selection for target detection in sar images," *Image Vision Comput. 21 (7)* (2003) 591–608.

[88] P. Viola, M. Jones, D. Snow, "Detecting pedestrians using patterns of motion and appearance," *IEEE International Conference on Computer Vision*, 2003.

[89] R. Collins, Y. Liu, "On-line selection of discriminative tracking features," *IEEE International Conference on Computer Vision*, 2003.

[90] N. Haering, N. da Vitoria Lobo, "Features and classification methods to locate deciduous trees in images," *Comput. Vision Image Understanding* 75 (1/2) (1999) 133–149.

[91] Y. Liu, J. Kender, "Video frame categorization using sort-merge feature selection," *IEEE Workshop on Applications in Computer Vision*, 2002, pp. 72–77.

[92] T.M. Cover and J.M. Van Campenhout, "On the Possible Orderings in the Measurement Selection Problem," *IEEE Trans. Systems, Man, and Cybernetics, vol. 7, no. 9*, Sept. 1977.

[93] Lindeberg, T., "Feature detection with automatic scale selection," *IJCV 30(2)* (1998).

[94] K. Rangarajan, M. Shah and D.V. Brackle, "Optimal corner detector," *Computing Vision, Graphics, and Image Processing*, Vol. **48**, 1989, pp. 230-245.

[95] C. Harris and M. Stephens, "A combined corner and edge detector," *the 4th Alley Vision Conference*, Manchester, 1988, pp. 147–151.

[96] J. F. Canny**,** "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI 8(6)*, Nov 1986, pp. 679–698

[97] L. Kitchen, A. Rosenfeld, "Gray level corner detector," *Pattern Recognition Letters*, 1982.

[98] Rafajlowicz, "SUSAN edge detector reinterpreted, simplified and modified", *International Workshop on Multidimensional (nD) Systems, Portugal, p.69-74, June 2007*, pp. 69-74.