

Chapter 1 Introduction

1.1 Research motivation

With the rapid growth of surveillance equipments, detecting, tracking and recognizing moving objects in the roadway videos is currently a popular issue. Because most traffic accidents are caused by cars and the main target in a criminal case is usually a fleeing car, cars are the most important moving objects in the captured frames. A vision based vehicle analysis system capable of identifying the sizes, models and colors of cars automatically would be useful for traffic monitoring and surveillance systems [1].

In vision based vehicle analysis systems, image processing and computer vision techniques are used to extract the features of captured vehicles. License plate numbers are the most commonly used features and the techniques to extract license plates are maturely developed. In [2], a few constraints on license plate recognition, such as fixed illumination, limited vehicle speed, designated routes, and stationary backgrounds had been considered. A number of fuzzy disciplines and neural subjects are defined to extract license plates and identify the digits. Because the researches of license plate recognition are abundant, Anagnostopoulos [3] provided a review to categorize and assessed them.

If the license plate numbers of a car are recognized, the appearance information, such as size, model and color about the car can be acquired from the database. However, license plates may be counterfeited or hidden, particularly in criminal cases. Because appearance features of vehicles are harder to be hidden or counterfeited than the license plates, vehicle feature analysis is more applicable than license plate recognition in vision based surveillance systems. Commonly analyzed car features include color and size of a car, as shown in Figs. 1(a) and (b). Car model indicates a type of cars marked by a certain car manufacturer and

appearing a unique shape. The model of a car can be identified by the appearance of the lamps and bumpers, or relative positions of the logos, as shown in Fig. 1(c).



Fig.1 (a) Some examples of black, gray, white, red, yellow, green, and blue cars, (b) small-size cars (first two), medium-size cars (middle two), and big-size cars (last two), (c) car models, including Mitsubishi lancer, Mazda 3, Nissan march, Honda civic, and Toyotayaris

Two challenges: non-homogeneous light reflection, pose variation and various colors of each car model are encountered when analyzing appearance features on real road vehicles. Fig. 2 shows two car images whose light reflection is non-homogeneous. The intensity values of pixels on car bodies are inconsistent due to highlight and shadows. The non-homogeneous light reflection seriously influences the segmentation of car body and further the classification of color. Fig. 3 shows car images with different orientations. Because the shape of a car varies dramatically, identifying a tested car with a template whose rotation angle is not exactly the same with the tested car is difficult.



Fig.2 Car images with non-homogenous light reflection.



Fig.3 Car images covering different vehicle orientations

1.2 Research objectives

The objective of this dissertation is to develop a traffic monitoring system whose capabilities are listed as follows:

- The system is able to identify appearance features, including sizes, models, and colors of captured vehicles. Three size classes are defined: small-size including sedans, medium-size composing of vans or pickups, and big-size including trucks and tourist buses. Ten color types are classified: black, gray, silver, red, yellow, orange, pink, purple, green, and blue. Furthermore, if a number of car models have been trained, the models of tested car can also be identified whether it belong to one of the trained models or none of them.
- The system is able to separate improper foreground and impure background from car body, further the color type of the car can be correctly classified. To acquire more accurate car body, the processes to resist the influence of non-homogenous light reflection are developed. For any color cars, the system is able to segment intact and complete car body and further determine accurate color category and color type.

- The camera of the traffic monitoring system is assumed to be established at a fixed position, but the captured cars may maneuver or drive through an open field. Hence, the viewpoint between the camera and the cars are of variety. The system should be able to analyze car features without horizontal viewpoint constraints.
- To employ the system in real road traffic surveillance, the computation load should be limited. The anticipated computation time of determining colors, classifying sizes and recognizing models of a car image whose resolution is 100×100 pixels are less than 60ms, 30 ms and 500 ms, respectively. To guarantee the efficiency of the system, the initial pose should be estimated under a very low computation load, for example, less than 5 ms.

1.3 Proposed solutions

The flowchart of the proposed traffic monitoring system is depicted as in Fig. 4. It comprises four main components: color determination, pose estimation, size classification, and model recognition. Each component is introduced briefly in this sub-chapter and described in greater details in Chapter 3, 4, 5, and 6, respectively.

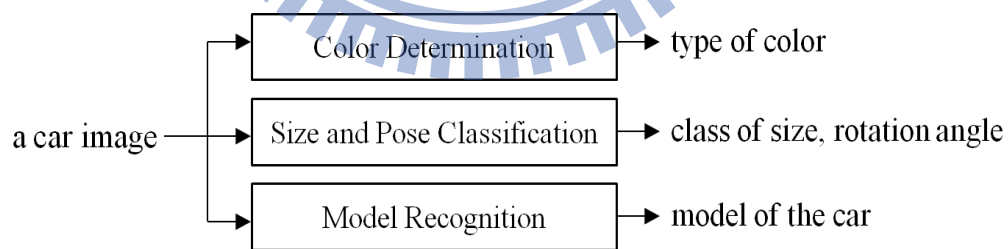


Fig.4 System overview

The color determination component is going to determine the color type of the tested car under non-homogeneous light reflection. Since the influences of light reflection on bright, dark and colored cars are different, a tri-states architecture algorithm that properly processes any color cars is proposed. Image segmentation is adopted to cluster regions sharing similar

color information. Then a SARM (Separating and Re-Merging) algorithm composes one or several regions to produce the car body. The algorithm carefully merges the regions depending on the color category that belongs to the car body, and the critical parameters for each category are properly selected. The proposed system is capable of managing both the difficulties of various viewpoints and non-homogeneous light reflection, thereby facilitating extraction of the car body, and enabling accurate classification of vehicle colors.

Since satisfactory performance for size classification and model recognition can be acquired if a sufficiently accurate pose is given. However, it is difficult to obtain a sufficiently correct initial pose because very little prior information is available in the initial phase. In the pose estimation component, an algorithm that extracts the symmetric center of the front/rear region of cars and computes the distance of the symmetric center to the nearest boundary is proposed. Two times of the distance between the symmetric center and the closest boundary of it is considered as the head (or rear) width. Since the head (or rear) width is bigger along with the increment of the rotation angle, the pose of the car can be estimated by the head (or rear) width. The algorithm provides sufficiently correct initial pose estimation without generating a heavy computation load. Though 3D images can generally obtain more accurate pose estimation, processing 2D images are faster and more practical for the real time systems.

The size classification component classifies the input tested car into three size categories: small-size, medium-size and big-size cars. By observing abundant small-size and large size cars in large orientations, a concavity property is observed. If the vehicle belongs to a small car with a large anticlockwise pose, a triangle consisting of background pixels can be found in the right top corner. If the vehicle belongs to a big car with a large anticlockwise pose, the triangle is founded in the left top corner. The component detects the concavity triangle in the left or right top corner to decide whether the vehicle belongs to small or big cars with a large pose. The remainder vehicles that cannot be distinguished by concavity are further classified by the aspect ratios of the car image. Two aspect ratios: car height to head width and head

width to car width, are designed. The support vector machine is trained and utilized to separate small, medium, and big sizes of vehicles based on the aspect ratios.

To improve the performance of model recognition in severe pose variation cases, a mirror morphing scheme is proposed in the model recognition component. The scheme produces the mirror image of the non-symmetric image of a tested car and then synthesizes the car image and its mirror image to form a symmetric representation. Based on knowledge concerning the inherent symmetry of the tested car, the correspondence relationship between the shape features found in the tested car image and those of its mirror image can be established. Then through the process of morphing, the synthesized image appears in a typical (front, rear or side) view, with the symmetric axis positioned precisely in the middle of the image. When the synthesized image of the car is matched against the template image for car model recognition, the factors influencing recognition accuracy, such as pose error and center bias are eliminated. Therefore, satisfactory recognition performance in situations of widely divergent poses can be obtained.

The remainder of the dissertation is organized as follows. Chapter 2 presents the related works. The proposed approaches on car color determination, car pose estimation, car size classification, and car model recognition are introduced in Chapters 3, 4, 5, and 6, respectively. Chapter 7 draws the conclusions and future works.

Chapter 2 Related Works

2.1 Current research on car color determination

Color is the most obvious appearance feature on a car, so many approaches were proposed for car color determination. Current vehicle color determination methods construct color histograms from car images before matching the color distribution with the templates. Kim et al. [4] determined the optimal bin numbers of the histogram of hue, saturation, and intensity axes to acquire optimal performance. In [5], a method based on vector matching was proposed and the color distribution in HSI color space was claimed to be more distinguishable than in the RGB or YCbCr color space. In [6], an example-based algorithm was proposed, which exhibited the potential to reduce the effects caused by lighting variations when the vehicle pose was restricted.

The Support Vector Machine (SVM) is also commonly used when combined with the color histogram methods. In [7], the dominant color on car image was extracted and then the SVM method was introduced to classify the extracted dominant color. In [8], a grid kernel method, which detects color information from not only a pixel but also its neighboring pixels, was proposed to achieve a superior recognition rate. However, these SVM related techniques usually encounter a considerable problem. A number of undesired pixels on the windshield, headlights or the background often interfere with the color histogram, thereby decrease the classification accuracy. Therefore, recent studies only considered the pixels that were included in the car body to calculate color histograms.

A car body comprises metallic parts, as follows: the hood, the roof, and the side panels (excluding the windshield, headlights, and wheels). The works of extracting the car body from a car image usually require a number of restrictions. In [9], the vehicle image was captured in

the front view. Subsequently, a fixed position rectangle, which was guaranteed to cover the most significant part of the car body, such as the hood of the car, was acquired. A homogeneous region composed of pixels with similar colors was extracted within the rectangle and the vehicle color was identified by only the pixels of this homogeneous region. In [10], car images were captured in the rear view. The significant part of the car body was also extracted and the vehicle color was identified by the pixels within the significant part.

The viewpoint restriction must be removed for wider and more realistic applications. In [11], the vehicle was assumed to be located near the center of the tested image and occupied most parts of the image. A number of removal rules were defined to reject the undesirable pixels. The remaining parts were considered the car body. The background subtraction technique [12] was adopted in this approach to generate an image that enclosed the vehicle. For a moving vehicle or known background cases, the contour of the car can be detected and the tested car can be placed near the center of the image. However, if the tested car is static and the background is unknown, it is more difficult to determine the correct car body.

The image segmentation techniques may be used to acquire the car body on a vehicle image without any viewpoint and location constraints. Each image is segmented into multiple regions, and the car body is composed of one or several regions. In a commonly used color-based image segmentation approach, [13] (and its previous paper [14]) selected a set of hills (local maximum) in the color histogram, and the pixels were grouped into the hills based on the minimal color difference. If the bin number of the histograms is determined optimally, all pixels that belong to one significant object may be grouped into one region. The significant object is subsequently extracted and the color of the object is classified.

Since the above approaches do not seriously consider the various effects of the non-homogeneous light reflection on different color cars, they are usually bothered with the effect of the non-homogeneous light. In [15], a specular-free processing was proposed to effectively overcome the non-homogeneous light reflection for general colored objects.

However, the main side effect of the technique is that the extracted car bodies and classified colors for grayscale cars are destroyed. Therefore, developing an algorithm which is capable of managing the difficulties of non-homogeneous light reflection for any color type cars is still challenging nowadays.

2.2 Current research on car pose estimation

Since traffic monitoring cameras are usually installed at heavy traffic intersections, they may capture images of vehicles at any orientation. However, recognizing cars at any orientation remains a challenging problem. Numerous current researches [16-17] can only analyze images taken from a particular viewpoint. In [16], a camera is installed in front of a parking lot entrance, so that every image would be captured in the front view or rear view. In [17], a camera is mounted in a car driving down a freeway, so most of the cars are captured from behind. Such limitations make the car model recognition systems impossible to deal with many general cases, for instance, cars maneuvering or driving through an open field.

To deal with the pose problem without viewpoint constraint, some researchers [18-21] have estimated pose by fitting a 3D model to a 2D/3D tested image iteratively. When the model converges, the tested image is matched with the converged model to recognize its identification. In [18], the pose and albedo map of a tested car is estimated, and then the illumination condition can be recovered by using spherical harmonics representation. The recognition performance can be improved by matching the reference image against the target image after re-tilting and re-lighting. In [19][20], the algorithms to estimate vehicle pose were developed in stereo camera systems. In [19], the iterative closest point algorithm is proposed to estimate the vehicle poses in stereo camera system. Compared with variety of distance metrics, the proposed algorithm yields a higher accuracy for yaw angle estimation. In [21], a metric 3D model composing of numerous local descriptors is constructed for each object.

RANSAC and Mean Shift algorithm is adopted to register the model to the tested object. The approach achieves 6-DOF pose estimation for complex objects in cluttered scenes.

If a sufficiently good pose of a tested car is given, it is able to achieve satisfactory performance for object matching or appearance feature acquisition. Considering the computation load, the pose should be estimated roughly or obtained by a heuristic method in the initial phase. Previous researches [19-20] compute a rectangle that bounds the possible vehicle pixels and use the size (aspect ratio) of the rectangle to find a proper initial pose. Real-time systems generally preclude complicated processing in the initial phase, and the number of iterations to estimate the initial pose in these complex algorithms must be limited. Thus, a sufficiently correct initial pose is necessary to allow the algorithms to converge rapidly. If the initial pose is dissimilar to the correct result, the iterative algorithm may never converge. However, it is generally difficult to obtain a sufficiently correct initial pose because very little prior information is available in the initial phase. In addition, the initial pose must be estimated under a very low computation load to guarantee the efficiency of ITS.

2.3 Current research on car size classification

Vehicle size classification is an essential step in real-time intelligent transportation systems. For example, for the automatic highway charging, the vehicle size is the basis to decide the charge. By integrating the vehicle size classification function into current license plate recognition (LPR) systems [1-3], the criminal cases in which large cars hanging on small car licenses can be detected. By combining vehicle size classification with vehicle matching [22], some models with unmatched sizes can be excluded in advance; therefore, higher recognition rate and lower computation load can be anticipated.

Current vehicle classification researches can be classified in three types: model-based, feature-based, and template matching approaches. In the model-based approaches [23-25], a few kinds of 2D/3D vehicle models are trained in advance. The models are iteratively adopted

to fit to the silhouette on the vehicle images. Among the converged models, the model with minimum difference is selected and the vehicle size is decided by the size of the selected model. In [24], when given two views of an object, Gao et. al. proposed an approach to determine whether the views are for the same or different object. Then a compact set of 3D models is utilized to provide geometry constraints and transfer appearance features for object matching across disparate viewpoints.

The advantage of model-based approaches is that not only the vehicle size, but also the appearance features, such as the positions of windshield and lamps are acquired. Furthermore, the model of the car can also be recognized if the number of training template is sufficient. However, model-based approaches usually require a long computation time to converge the iteration of 3D model. Hence, these methods are not applicable to real-time intelligent transportation systems.

In the feature-based approaches [26-29], the silhouette of the vehicle is extracted by background subtraction [30-31] or frame difference techniques [32-33]. A bounding box to enclose the silhouette is generated and then some shape features, such as area size [27], estimated length [24], ratio of height to width [29], and solidity of the bounding box are computed. A classifier, for example, support vector machine (SVM), is commonly utilized to classify the vehicle size [29] based on the shape features. These feature-based methods are able to rapidly provide classification results. However, the shape features change significantly along with the increment of the rotation angles. Therefore, in the current feature-based vehicle classification studies, the vehicle pose is usually fixed or known [26-29].

Other researches [34-37] collected abundant size templates and then adopted matching technique, such as SAD (sum of absolute difference), NCC(normalized cross-correlation), or edge-based methods to decide to which template the tested car is most similar. The size of the most similar template is considered the size of the tested car. In [35], a quasi-rigid alignment combined with a flexible matching method is proposed. The tested car is aligned to the

template car based on the established corresponding relationships between the key segments on the tested car and the template car. Subsequently, the local flexible matching mechanism divides the tested car into multiple patches. Each patch is shifted in the neighboring regions to obtain the optimal match to the corresponding part of the template car. In [36-37], an active shape model (ASM) is described. The model is built by learning patterns of variability from a training set of correctly annotated images. After aligning the shape of the model to the same shape of the tested car, the center bias of the shape feature is reduced; hence, the tested car is able to correctly be matched with the template car.

2.4 Current research on car model recognition

Since the appearances of vehicles in different size categories are quiet different, it is easier to identify car size than car models. Vehicles in the same size class tend to appear similar, so template matching is preferred to be model-based and feature-based approaches to recognize the model of a tested car. Since the pose variation deforms the shape of the car appearance dramatically as shown in Fig. 3, the template matching methods usually suffer from the pose variation problem. Pose problem is a thorny issue in most object recognition fields; hence, many current approaches [16-17] of car recognition are applicable only from a particular viewpoint. To enable car recognition without viewpoint constraints, some researchers [38-39] estimate the pose first and then match the image to a specific template with the same pose as the estimated one.

In [38], a method for identifying ellipses and matching the ellipse in 2D to a wheel in 3D was proposed. By estimating the normal of the ellipse in 3D space, the rotation angle of the vehicle can be acquired. In [39], the vehicle pose is estimated in every frame by calculating its 3D motion on a static plane. Then a number of 3D models are selected from the database and all the models are rotated to the same pose as the tested car on the frame. The model with the best matching score is identified as the model of the tested car.

One drawback of these approaches based on pose estimation is that slight deviation in pose estimation degrades the recognition performance dramatically. Nonetheless, it is difficult to estimate the pose of a car with a high degree of accuracy. In [35], a quasi-rigid alignment combined with a flexible matching method is proposed to deal with pose variation for car identification. This method identifies cars correctly if they have distinctly different shapes or colors. However, it still has the difficulty in discriminating a car from other cars with similar shapes and colors. Therefore, finding an effective algorithm for car model recognition to accommodate the severe pose variation problem in the cases of similar shape and color would be of great value and interest.

In previous studies, symmetry was used to track, locate, or recognize vehicles [40-41] by identifying the central line of tested cars. Subsequent researches [42-43] improved the means by which symmetry is measured. Since these methods rely heavily on “explicit” symmetric properties found in the images of cars, they are generally applicable only within the narrow orientation of front or rear view. When a car is orientated in a wider view, the explicit symmetry may be attenuated in the car image. Therefore, in severe pose orientation situations, such symmetry-based methods are barely able to differentiate one car from other cars with a similar shape and color.

Chapter 3 Car Color Determination Algorithm

To resist the influence of non-homogenous light reflection on car color determination, an algorithm [44] consisting of tri-states architecture is proposed. The algorithm properly processes the tested cars in colored, bright, and dark categories. The flowchart of the algorithm is shown in Fig. 5, and the terms utilized in this figure are explained in Table 1. The algorithm comprises three major steps: car body candidate generation, car body determination, and vehicle color classification. Chapter 3.1 presents an overview of the algorithm. The three major steps are introduced in Chapters 3.2, 3.3, and 3.4, respectively. Finally, the experimental results are provided in Chapter 3.5.

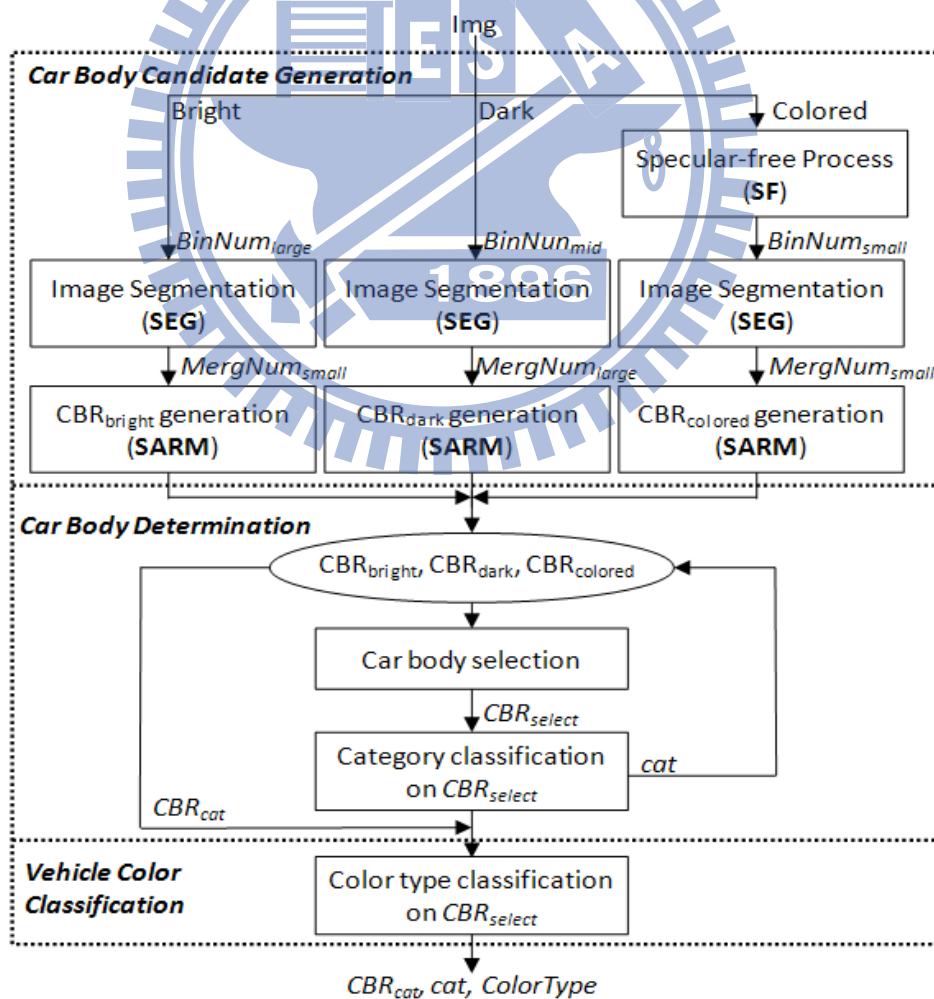


Fig. 5 The framework of the car body extraction and color classification mechanism

Table 1 Terminology table of the car color determination algorithm

Terms	Definition
<i>Img</i>	an input vehicle image
<i>BinNum</i> _{large, mid, small}	predefined small, medium and large bin numbers of histogram
<i>MergNum</i> _{large, small}	predefined small and large merge times
<i>CBR</i> _{bright, dark, colored}	car body candidates for bright, dark or colored categories
<i>CBR</i> _{select}	the selected car body among the candidates
<i>cat</i>	the classified category on <i>CBR</i> _{select}
<i>CBR</i> _{cat}	the final extracted car body based on current category
<i>ColorType</i>	the final classified color type of the input vehicle

3.1 Algorithm overview

In Fig. 5, the input of the algorithm is a car image (*Img*) that may be any color type. The algorithm extracts the car body region (*CBR*_{cat}) and distinguishes the color category ($cat \in \{\text{colored, bright, dark}\}$) and the color type ($ColorType \in \{\text{white, gray, black, red, orange, yellow, green, purple, blue, pink}\}$) of the car. The algorithm comprises three major steps: car body candidate generation, car body determination, and vehicle color classification.

By observing numerous real vehicle images, we found that the influence of light reflection differs on colored, bright and dark vehicles. For example, under light reflection, RGB values of colored and dark cars would alter much more than those of bright cars. In addition, because light reflection generally originates from strong bright luminance, light reflection for colored cars can be distinguished more easily from the chromaticity of the car body. Therefore, this study proposes tri-states architecture for extracting an intact car body under different light situations. Three car body candidates are generated simultaneously for the three color categories. The tested car image is optimally processed for each color category, and finally, one of the candidates is determined as the extracted car body.

To generate the three candidates, an image segmentation technique [13] based on color histogram is employed first to segment the input image into multiple color coherent regions.

Generally, the number of the segmented regions is proportional to the bin number ($BinNum$) of color histogram. If the bin number is too small, a number of regions in the background are easily included in the extracted car body. However, when the bin number is too large, the car body region is split into numerous small pieces, and some of them may be excluded from the extracted car body. To acquire preferable segmentation, different bin numbers should be given for different categories.

For colored cars, because the effect of light influence is variable, a specular-free processing is adopted before segmenting the image. The specular-free process [15] uses the chromaticity information to generate a specular-free image. Subsequently, the specular-free image is segmented by adopting a small number of color bins ($BinNum_{small}$) and a segmented image is produced. For bright and dark cars, since no chromaticity information exists in the car body, the specular-free processing is not applied to distinguish the intensity of the car body. Large and medium bin numbers ($BinNum_{large}$, $BinNum_{mid}$) are required for bright cars and dark cars individually, to prevent the car body from grouping with the background. However, more small pieces are generated when larger bin numbers are adopted. Therefore, a merging process is necessary to recover the whole car body.

To generate the car body candidates (CBR_{bright} , CBR_{dark} , and $CBR_{colored}$) more accurately, a SARM (Separating and Re-Merging) algorithm is proposed. The algorithm considers not only the color feature but also the spatial relationships between each pairs of segmented regions on vehicle images. Since the influences of light reflection on bright, dark and colored cars are various, different values are assigned to the number of merging times. For dark cars, a large merging number ($MergNum_{large}$) is given, since more small pieces are produced after segmentation. For colored and bright cars, the influence of light reflection will not make so many small pieces as in the dark car cases. Thus, a small merging number ($MergNum_{small}$) is sufficient.

When all car body candidates: CBR_{bright} , CBR_{dark} , $CBR_{colored}$, for the three categories have been prepared, a selection scheme that compares the significance and compactness of each candidate is proposed. After processing this scheme, a car body (CBR_{select}) is selected from the candidates and the color category of the selected car body is classified by SVM technique. Based on the classified category, the candidate (CBR_{cat}) which has identical category to the classified result is obtained as the extracted car body.

After the car body of the input car has been extracted, the color type ($ColorType$) of the vehicle is decided only according to the pixels in the extracted car body. A hierarchically structured SVM is adopted in this paper. In the first layer, a binary SVM is utilized to classify the vehicle into colored or grayscale class. If the vehicle belongs to the grayscale class in the first layer, a multi-class SVM is designed to classify the image into a black, gray, or white car in the second layer. If the vehicle is identified to belong to the colored category in the first layer, another multi-class SVM is designed to classify the image into a red, orange, yellow, green, blue, purple or pink car in the second layer.

3.2 Car body candidate generation

3.2.1 Image segmentation

To segment a vehicle image into multiple regions with coherent color attributes, abundant image segmentation techniques have been proposed. Among these studies, a color-based image segmentation algorithm: hill climbing [13-14] is adopted, due to its sufficient effectiveness and efficiency for application in real-time transportation systems.

The color histogram in the CIELAB color model [45] is computed and a set of local peaks on the histogram are searched. The adjacency pixels sharing similar colors to the color of the peak are grouped together and considered a color coherent segment. The number of

peaks calculated via hill-climbing algorithm is under control by the number of histogram bins; hence, influencing the number of segments ($SegNum$) is possible by adjusting the bin number ($BinNum$). After image segmentation, a segmented image ($SImg$) is generated and composed of multiple regions ($Reg^1, \dots, Reg^{SegNum}$), as Eq. (1).

$$SImg = \{ Reg^i \mid 1 \leq i \leq SegNum \} = SEG(Img, BinNum) \quad (1)$$

Fig. 6(a)-(c) show a white, a black, and an orange car images. The segmented results via the hill-climbing algorithm are shown in Fig. 6(d)-(f). The pixels grouped together in the color histogram are represented by a unique color. The color coding scheme is based on the number of regions and the built-in MATLAB colormap “jet” [46]. If the number of region is $SegNum$, $SegNum$ colors selected from the colormap are acquired to draw the image.

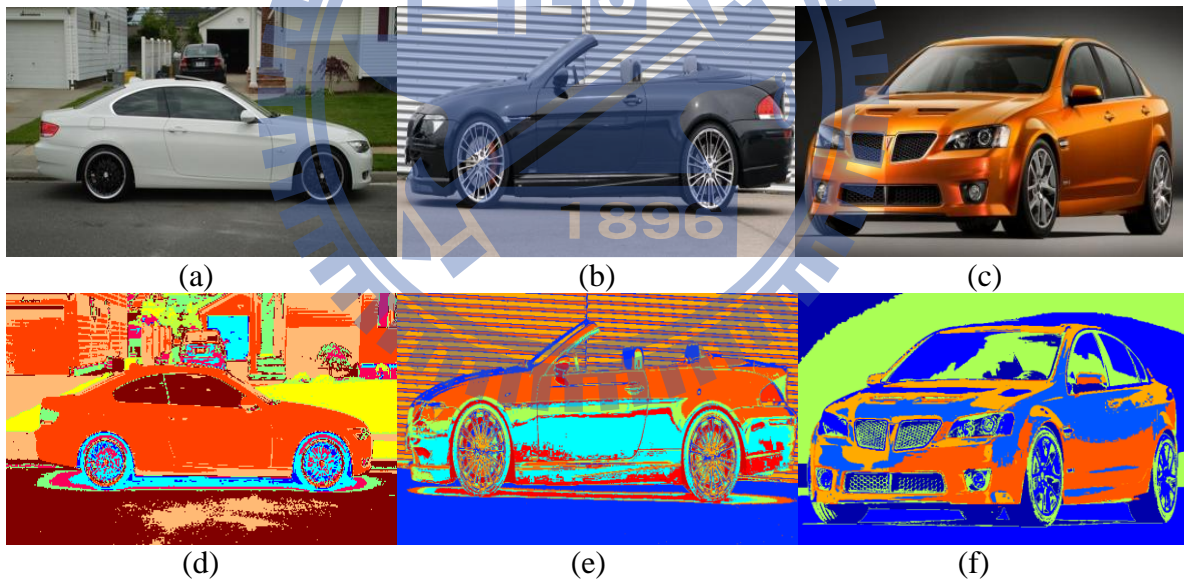


Fig. 6 (a) A white car image, (b) a black car image, (c) an orange car image. The image segmentation results by hill-climbing of the (d) white, (e) black, and (f) orange car image

From 6 (d)-(f), some problems can be found. First, some background pixels are misclassified into the car body. Second, the car body may be disrupted into multiple large partitions. Third, some small fragments appear on the car body due to light reflection and these fragments have extremely different colors to the car body.

3.2.2 Specular-free processing

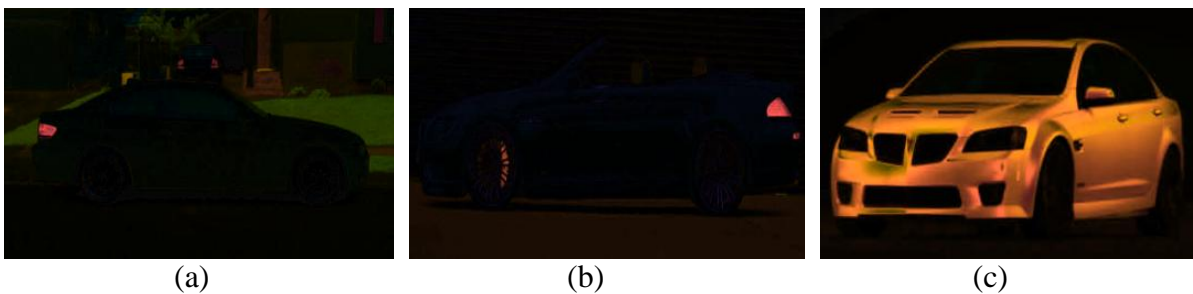
The colored cars can be processed via a specular-free operator to alleviate the influence of light reflection. The luminance reflected from an object (including cars) can be formulated by a linear combination of diffuse and specular reflections:

$$Img_c(x) = m_d(x) \times A_c(x) + m_s(x) \times \Gamma_c \quad (2)$$

where $Img_c(x)$, $m_d(x)$, $m_s(x)$, $A_c(x)$, and Γ_c are the intensity value in RGB color space (i.e., $c \in \{r, g, b\}$), the diffuse coefficient, the specular coefficient, the diffuse chromaticity, and specular chromaticity of a pixel x on an image Img .

The diffuse reflection term $m_d(x) \times A_c(x)$ reveals the actual color of the object, while the specular reflection term $m_s(x) \times \Gamma_c$ originates from the undesirable light reflection. Numerous studies [15, 47-48] have developed methods for separating the diffuse and specular reflections. This study adopts a mechanism [15] capable of transforming the specular image into the specular-free image (i.e., $SF_c(x) = m_d(x) \times A_c(x)$) to overcome the influence of light reflection in the car body extraction problem.

To construct a specular-free image, a luminance-normalized image with pure white specular components is produced. A non-linear shift function is computed to transform the intensity and chromaticity of each pixel to the normalized image while retaining the color information. Fig. 7(a)-(c) show the specular-free images derived from Fig.6(a)-(c).



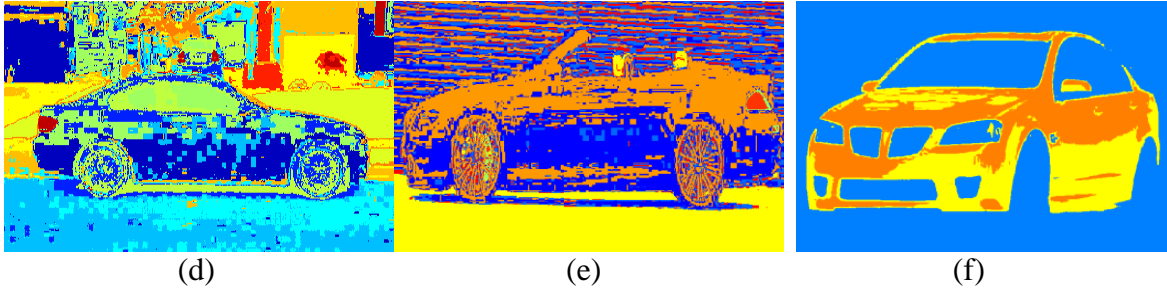


Fig. 7 (a)-(c) The specular-free images of the white, black, and orange cars, (d)-(f) the segmented results of the specular-free images via hill-climbing algorithm

In Fig. 7(a) and 7(b), specular-free processing is unsuitable for application on grayscale (dark or bright) cars. Since the diffuse components of all grayscale pixels approximate to zero, the car body is severely destructed and becomes ever more difficult for segmentation. Hence, this study only applies specular-free processing for colored cars. In Fig. 7(c), a specular-free processing is performed on the orange car. The segmented result of Fig. 7(c) is shown in Fig. 7(f). Since the influence of light reflection has been evidently alleviated, the car body of the orange car (and also for other colored cars) can be segmented more effectively.

3.2.3 Separating and re-merging

The traditional car body segmentation research [11] selects the significant segmented region as the car body. However, the significant part of the vehicle image may include impure background pixels (environment) or improper foreground pixels (windshield, lights, wheels, etc); therefore the segmented regions must be “separated”. Because the car body may be disrupted into several large partitions or small fragments, a “merging” process must be performed to recover the entire car body. The large partitions and the small fragments usually have various characteristics. For example, a small fragment incurred by the strong light reflection generally has considerably high intensity. Whereas, a large partition caused by environmental luminance usually has the intensity value close to that of the car body. The merging strategies of small fragments and large partitions must be designed differently.

Three functions for large partition separating, small fragment merging, and large partition merging are proposed, which are referred to *partitionSeparate*, *microMerge*, and *macroMerge*, respectively. These functions are integrated into the SARM (Separating and Re-Merging) algorithm. To separate the background and merge multiple partitions into a car body, the algorithm simultaneously considers color feature and spatial relationships and properly controls the parameters. The procedures of these functions are described as follows.

Function *partitionSeparate*: This function separates the disconnected partitions of each consistent color region in the original segmented image (*SImg*) produced by the hill-climbing algorithm. Because the algorithm segments images only considering the color features, the pixels with a similar color are clustered regardless of whether they are far away from each other. This function considers the spatial relationships between the regions to generate a new segmented image (*SepImg*). It contains three steps as shown in List 1.

In step 1, the regions in the input segmented image, the sizes of which are larger than a size-threshold ($th_{size} = 200$ pixels, empirically), are collected; otherwise, the small regions are not separated. In step 2, for each collected region, the 8-connectivity is accessed by the algorithm (*8-ConnectivityCheck*) in [49]. After the process of *8-ConnectivityCheck*, each region (Reg^i) is separated into k partitions (Par_1^i, \dots, Par_k^i). In step 3, each partition is considered as a new region, and assigned a distinct label. Finally, the new labeled regions ($SepReg^1, \dots, SepReg^m$) compose of a new segmented image (*SepImg*).

Procedure: $[SepImg] = \text{partitionSeparate}(SImg, th_{size})$.

Input: $SImg = \{Reg^1, \dots, Reg^n\}$: a segmented image created by hill-climbing and composed of a set of segmented regions; the number of regions is n

th_{size} : a size threshold to determine large and small regions

Output: $SepImg = \{SepReg^1, \dots, SepReg^m\}$: a new segmented image composed of more segmented regions; the number of regions is m ($m \geq n$)

Step

0 $m = 0$; // initialization of the number of new segmented regions

For $1 \leq i \leq n$

1 **If** $\text{size}(Reg^i) > th_{size}$

// Separating the region into k partitions by checking 8 neighboring
// pixels of each pixel in the region.

2 $[Par_1^i, \dots, Par_k^i] = \text{8-ConnectivityCheck}(Reg^i)$;

3 **For** $1 \leq j \leq k$

$SepReg^{m+j} = Par_j^i$;

End

$m = m + k$;

Else

$m = m + 1$; $SepReg^m = Reg^i$;

End

End

List 1 The procedure of *partitionSeparate* function

Function *microMerge*: This function merges the small size regions on the input segmented image (*SepImg*) and creates a new segmented image (*MergSepImg*) only containing sufficiently large regions. The function contains four steps, which are described in List 2. In step 1, the segmented regions are classified into small-size fragments ($smallFrag^1, \dots, smallFrag^{smallCount}$) and large-size partitions ($MergSepReg^1, \dots, MergSepReg^{largeCount}$) by comparing their areas with th_{size} . In step 2, for each pair of small-size fragment ($smallFrag^i$) and large-size partition ($MergSepReg^j$), check they are appearing in the neighboring area or not. If they are neighbored, the function NeighborTest returns 1; otherwise, returns 0.

In step 3, the hue, saturation and intensity differences ($diff_{hue}$, $diff_{sat}$, $diff_{int}$) between the small fragment and all of its neighboring large partitions are computed. Among the large partitions whose color differences are all smaller than the corresponding thresholds (th_{hue} , th_{sat} , th_{int}), the largest partition is selected. The small fragment is merged into the selected large partition. The thresholds differ on categories because the light reflection influence on dark

cars differs from that of the white or colored cars. The thresholds for each category are discussed in Chapter 3.1.4. Finally, the large merged partitions compose of the new segmented image ($MergSepImg$).

```

Procedure: [ $MergSepImg$ ] = microMerge( $SepImg$ ,  $th_{color}$ ).
Input:  $SepImg = \{SepReg^1, \dots, SepReg^m\}$ : the input segmented image and its regions
         $th_{color} = [th_{hue}, th_{sat}, th_{int}]$ : difference thresholds of hue, saturation, and intensity
Output:  $MergSepImg = \{MergSepReg^1, \dots, MergSepReg^{largeCount}\}$ : the segmented image after merging
        small regions into large ones; the regions number is  $largeCount$ 

Step
1  For  $1 \leq i \leq m$ 
    If ( $size(SepReg^i) < th_{size}$ )  $smallCount++$ ;  $smallFrag^{smallCount} = SepReg^i$ ;
    Else  $largeCount++$ ;  $MergSepReg^{largeCount} = SepReg^i$ ;
    End
  End
2  For  $1 \leq i \leq smallCount$ 
     $max\_size = 0$ ;
    For  $1 \leq j \leq largeCount$ 
      // determine the small fragment  $smallFrag^i$  and the large partition  $MergSepReg^j$ 
      // whether they are appearing in the neighboring ( $7 \times 7$ ) area or not
      If  $NeighborTest(smallFrag^i, MergSepReg^j) == 1$ 
3    [ $diff_{hue}, diff_{sat}, diff_{int}$ ] =  $ColorDiff(\text{avg}(smallFrag^i), \text{avg}(MergSepReg^j))$ ;
      If ( $diff_{hue} < th_{hue} \& diff_{sat} < th_{sat} \& diff_{int} < th_{int}$ ) & ( $size(MergSepReg^j) > max\_size$ )
         $max\_size = size(MergSepReg^j)$ ;
         $max\_index = j$ ;
      End
    End
  End
   $MergSepReg^{max\_index} = MergSepReg^{max\_index} \cup smallFrag^i$ ;
End

```

List 2 The procedure of *microMerge* function

Although these fragments are merged into the large partitions, their colors are not counted to the average color of the large partition. The colors of the merged fragments are adapted to be equal to the average color of the large partitions which those fragments are merged.

Function *macroMerge*: This function selects one or multiple regions from the new segmented image ($MergSepImg$) to compose the car body (CBR_{cat}). The function consists of four steps and is described in List 3.

In step 1, a connection table is constructed which describes the connectivity of each pair of the large partitions. For each boundary pixel in each large partition, check the other large partitions whether appearing in the neighboring (7×7 pixels) area. If two large partitions have over $num_{neighbor}$ pixels (empirically, $num_{neighbor} = 60$) appearing in the neighboring area, these two partitions are considered to be connected. The connectivity relationships between all pairs of the partitions in *MergSepImg* are recorded in the connection table.

In step 2, a seed region that will merge the other regions is selected. In general, the central part of an image is more significant than the boundary part [50]. Therefore, the image is firstly divided into the following two parts: inner part (*InnerImg*), and outer part (*OuterImg*) by two parameters δ_x , δ_y . The parameters are defined as a fraction (for example, 1/10) of the width and the height of the image. The area with width ranging from $[\delta_x, wid - \delta_x]$ and height ranging from $[\delta_y, len - \delta_y]$ is considered as the inner part, and the remaining area is considered as the outer part. For each segmented region, the pixel number in the inner part (*InPxNum*) and the number in the outer part (*OutPxNum*) are both computed. If a region which its pixels in the outer part are more than a predefined threshold ($th_{OutPxNum}$) or its ratio of *OutPxNum* over *InPxNum* is greater than another predefined threshold ($th_{OutInRatio}$), the region is considered as the background. Subsequently, the number of the pixels in the inner part is ignored and set to zero. Then, this region will not be selected as the seed region.

Furthermore, the average saturation of each region is computed and used to decide the seed region. If the input category (*cat*) is bright or dark, the regions with high saturation (i.e., larger than a predefined threshold $th^{upbound}_{sat}$) are not selected as the seed region and their inner pixel numbers are set to zero. If the input category is colored, the regions with low saturation (i.e., lower than a predefined threshold $th^{lowbound}_{sat}$) are not selected as the seed region and their inner pixel numbers are set to zero. Next, the top three largest partitions in *MergSepImg* and passed the above tests are selected as the seed candidates. If the category is bright, the candidate which has the highest average intensity is selected as the seed region. If

Procedure: $[CBR_{cat}] = \text{macroMerge}(\text{MergSepImg}, th_{color}, \text{MergNum}, cat)$.

Input: $\text{MergSepImg} = \{ \text{MergSepReg}^1, \dots, \text{MergSepReg}^{largeCount} \}$: the input segmented image

$th_{color} = (th_{hue}, th_{sat}, th_{int})$: the color difference threshold

MergNum : the number of iteration for merging

cat : the current category

Output: $CBR_{cat} = \{ CBR_{cat}^i, \dots, CBR_{cat}^k \}$: one or multiple regions selected as the car body

Step

```

1   ConnectTable = ConstructConnectTable (MergSepImg);
2   For 1 ≤ i ≤ largeCount
      [InnerImg, OuterImg] = DevideInnerOuter(MergSepImg);
      InPxNum[i] = # pixels ∈ MergSepRegi ∩ InnerImg ;
      OutPxNum[i] = # pixels ∈ MergSepRegi ∩ OuterImg ;
      If (OutPxNum[i] > thOutPxNum)           InPxNum[i] = 0;      End
      If (OutPxNum[i] / InPxNum[i] > thOutInRatio)   InPxNum[i] = 0;      End
      If (cat == 'bright' & avgSat(MergSepRegi) > thupboundsat)   InPxNum[i] = 0;
      Elseif (cat == 'dark' & avgSat(MergSepRegi) > thupboundsat)   InPxNum[i] = 0;
      Elseif (cat == 'colored' & avgSat(MergSepRegi) < thlowboundsat)   InPxNum[i] = 0;
      End
    End
    If cat == 'bright'
      SeedCandidates = TopThreeMax(InPxNum);
      seed = MaxIntensity(SeedCandidates);
    Elseif cat == 'dark'
      SeedCandidates = TopThreeMax(InPxNum);
      seed = MinIntensity(SeedCandidates);
    Else //cat == 'colored'
      seed = TopOneMax(InPxNum);
    End
    InPxNum[index(seed)] = 0; // clear inner pixel number of the seed region
3   iter_count = 0; MacroMergList = Null; list_num = 0;
    Do maxPartIdx = maxIndex(InPxNum); // get the index of max region in InPxNum
      iter_count ++;
      [diffhue, diffsat, diffint] = ColorDiff (avg(seed), avg(MergSepRegmaxPartIdx));
      If diffhue < thhue & diffsat < thsat & diffint < thint
        If ConnectTable[seed, MergSepRegmaxPartIdx] == 1
          CBRcat = seed ∪ MergSepRegmaxPartIdx;
        For 1 ≤ i ≤ list_num
          If ConnectTable[MacroMergListi, MergSepRegmaxPartIdx] == 1
            CBRcat = seed ∪ MacroMergListi;
            MacroMergList = MacroMergList - MacroMergListi;
          End
        End
      Else list_num ++;
      MacroMergList = MacroMergList ∪ MergSepRegmaxPartIdx;
    End
    InPxNum[maxPartIdx] = 0;
4   While iter_count < MergNum & size(MergSepImgmaxPartIdx) > 0

```

List 3 The procedure of *macroMerge* function

the category is dark, the candidate which has the lowest average intensity is selected as the seed region. If the category is colored, the candidate with the largest area is selected as the seed region.

In step 3, the partition ($MergSepReg^{maxPartIdx}$) with maximal area is selected and excluded at the end of loop. The connectivity and average color of the partitions are assessed with the seed region. If the color difference between the partition and the seed is smaller than the thresholds (th_{hue} , th_{sat} , th_{int}), the connectivity between them are further assessed. The partition is merged into the seed region if it is connected to the seed. Otherwise, it is added into a merge list ($MacroMergList$). All of the partitions in the merge list are assessed when any new partition is merged into the seed region. If the connectivity between any partition in the merge list and the new partition ($MergSepReg^{maxPartIdx}$) exists, the partition in the merge list is excluded from the merge list and merged into the seed region. In step 4, the number of iteration ($iter_count$) and the input parameter $MergNum$ are assessed. If they are equal, the function returns the merged result. Otherwise, step 3 is repeated.

The car body of the tested car can be recovered more completely through the processing of the SARM algorithm. Fig. 8 shows the segmentation results after the SARM algorithm processing. A number of weaknesses, as shown in Figs. 6 and 7, are effectively overcome.

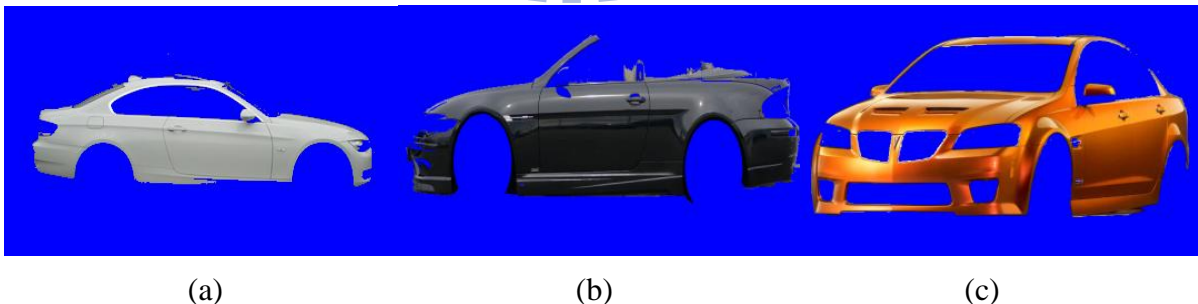


Fig. 8 The extracted car body candidates of the (a) white, (b) black, and (c) orange cars after the process of the SARM algorithm

3.2.4 Tri-states car body candidate generation

In List 4, the input data comprised the following: (1) a vehicle image (Img), (2) the category of current state: $cat \in \{bright, dark, colored\}$, (3) the bin number of color histogram ($BinNum$) for each category, (4) the merging times ($MergNum$) of **macroMerge** for each category, and (5) several predefined thresholds (th_{color}) for each category. The output of this process is an extracted car body region (CBR_{cat}) depending on the current category (cat). The procedure consists of five steps described as follows.

In step 1, if the category is colored, the specular-free operator (**SF**) is executed on the input image (Img) and a specular-free image is produced. This operator is not performed in the bright or dark categories. In step 2, the output image of step 1 is segmented by the function **SEG**, as described in Chapter 3.1, and a segmented image ($SImg$) is produced. Because the bin number of the histogram ($BinNum$) affects the capability of separating the car body and the background, this parameter should be set carefully.

In step 3, the function **partitionSeparate** is executed to generate a new segmentation image ($SepImg$) which clusters the pixels by not only considering the color but also the connectivity relationship. The disconnected partitions with the consistent color region in the original segmented image ($SImg$) are separated. In step 4, the function **microMerge** is invoked to prevent the generation of numerous mini regions in the new segmented image ($SepImg$). The mini regions in $SepImg$ are merged with their neighboring larger regions and a new segmented image ($MergSepImg$) is generated. In this step, a very large value is set to the intensity threshold ($microTH_{int}^{dark}$) for dark cars to resist the substantial influences of strong light reflection. For bright and colored cars, the medium values are assigned to the saturation threshold and the intensity threshold ($microTH_{sat}^{bright}$, $microTH_{int}^{bright}$) or ($microTH_{sat}^{colored}$, $microTH_{int}^{colored}$) so that the mini region will not be merged to the large region whose color is obviously different to the color of the mini region, even they are neighboring.

Procedure: Tri-states Car Body Candidate Generation Process**Input:** Img : an input vehicle image cat : category among bright/dark/colored $BinNum_{\{bright, dark, colored\}}$: the bin numbers of color histogram for each category $MergNum_{\{bright, dark, colored\}}$: the number of merging times for each category $microTH^{\{bright, dark, colored\}}_{\{hue, sat, int\}}$: hue, saturation and intensity thresholds for **microMerge** $macroTH^{\{bright, dark, colored\}}_{\{hue, sat, int\}}$: hue, saturation and intensity thresholds for **macroMerge****Output:** CBR_{cat} : the extracted car body region depending on the input category cat **Step**

```
1  If   $cat == 'colored'$        $I = SF(Img);$ 
   Else       $I = Img;$ 
2  If   $cat == 'colored'$        $BinNum = BinNum_{colored};$ 
   Elseif  $cat == 'bright'$      $BinNum = BinNum_{bright};$ 
   Else       $BinNum = BinNum_{dark};$ 
    $SImg = SEG(I, BinNum);$ 
3   $SepImg = partitionSeparate(SImg);$ 
4  If   $cat == 'colored'$        $th_{color} = [microTH^{colored}_{hue}, microTH^{colored}_{sat}, microTH^{colored}_{int}];$ 
   Elseif  $cat == 'bright'$      $th_{color} = [microTH^{bright}_{hue}, microTH^{bright}_{sat}, microTH^{bright}_{int}];$ 
   Else       $th_{color} = [microTH^{dark}_{hue}, microTH^{dark}_{sat}, microTH^{dark}_{int}];$ 
    $MergSepImg = microMerge(SepImg, th_{color})$ 
5  If   $cat == 'colored'$        $th_{color} = [macroTH^{colored}_{hue}, macroTH^{colored}_{sat}, macroTH^{colored}_{int}];$ 
    $MergNum = NumMerg_{colored};$ 
   Elseif  $cat == 'bright'$      $th_{color} = [macroTH^{bright}_{hue}, macroTH^{bright}_{sat}, macroTH^{bright}_{int}];$ 
    $MergNum = MergNum_{bright};$ 
   Else       $th_{color} = [macroTH^{dark}_{hue}, macroTH^{dark}_{sat}, macroTH^{dark}_{int}];$ 
    $MergNum = MergNum_{dark};$ 
    $CBR_{cat} = macroMerge(MergSepImg, th_{color}, MergNum, cat);$ 
```

List 4 Tri-states Car Body Candidate Generation Process

In step 5, the function **macroMerge** is called to construct the whole car body CBR_{cat} . A seed region is selected first, and one or more partitions in the segmented image ($MergSepImg$) are selected and merged to the seed region if they are connected and have similar colors. The thresholds for color difference on each category are settled differently. For bright or dark cars, a large intensity threshold (empirically, $macroTH^{bright}_{int} = macroTH^{dark}_{int} = 90$, if hue, saturation, and intensity values are ranged from 0 to 255) and a small saturation threshold (empirically, $macroTH^{bright}_{sat} = macroTH^{dark}_{sat} = 30$) are settled to collect most car body partitions and not to include the background regions with obvious colors. A considerably large value is assigned to the hue threshold (empirically, $macroTH^{bright}_{hue} = macroTH^{dark}_{hue} =$

255), because the hue value of grayscale color is meaningless. For colored cars, the saturation and hue values are more meaningful than the intensity value. Hence, the small values are defined for $macroTH^{colored}_{hue}$ and $macroTH^{colored}_{sat}$, and a considerably large value is set to $macroTH^{colored}_{int}$ (e.g. $(macroTH^{colored}_{hue}, macroTH^{colored}_{sat}, macroTH^{colored}_{int}) = (20, 30, 255)$). In this step, the number of the merging times ($MergNum$) must also be properly controlled for dark, bright, and colored cars individually to recover the entire car body.

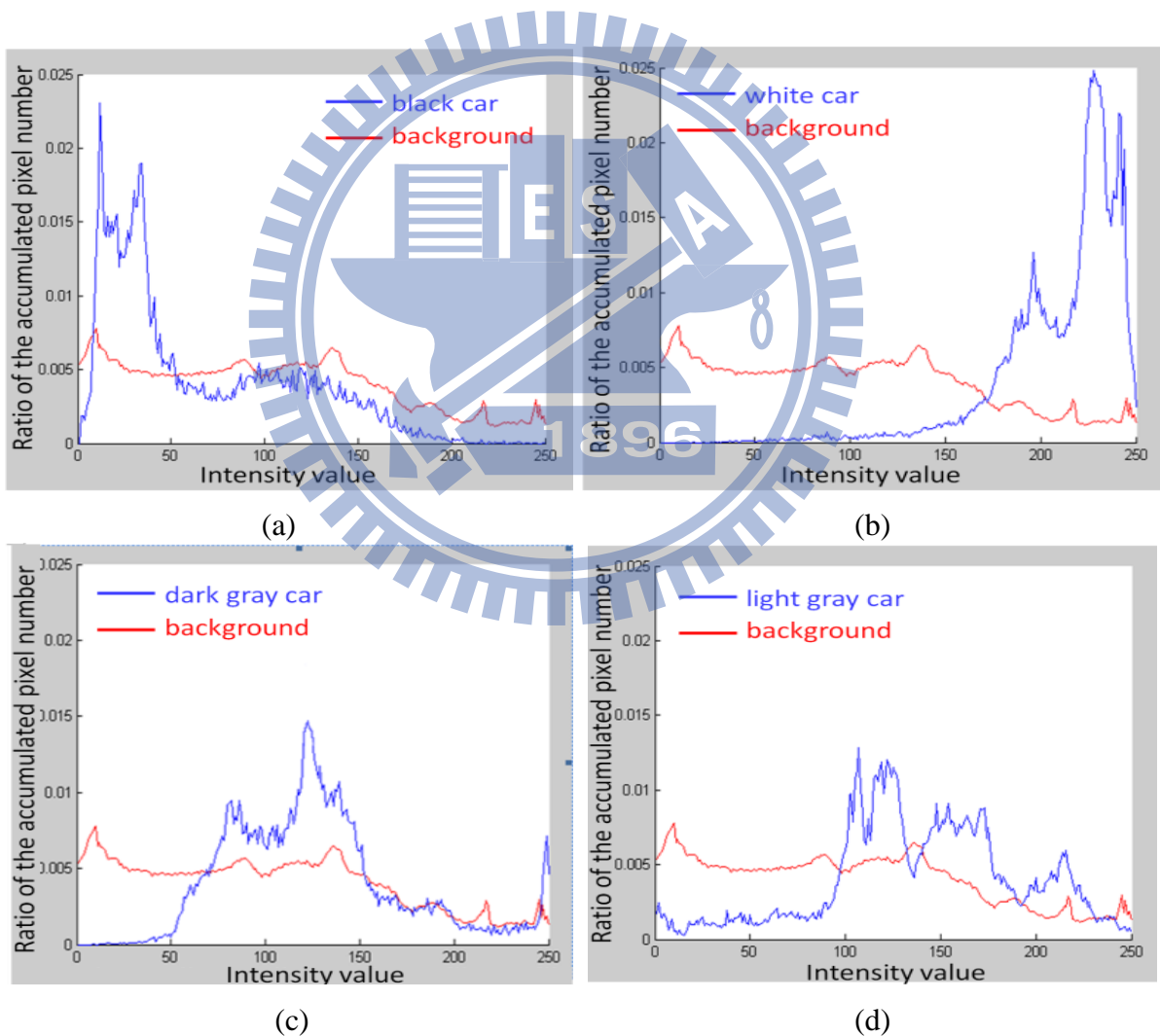
3.2.5 Process justification and parameter selection

In this subchapter, process justification and parameter selection of the tri-states architecture are described based on the characteristics of the color histograms and the spatial relationship of the background and cars of various colors. Two critical processes: (1) the strategy of selecting the seed region and (2) the tri-states architecture are justified. We also determine two key parameters: (1) the number of the color bins ($BinNum$) and (2) the number of macro merging times ($MergNum$), according to the requirements of optimization.

In Fig. 9(a)-(f), the color histogram distributions of six cars (blue lines) with black, white, dark-gray, light-gray, red, and yellow colors are demonstrated. Because the grayscale cars can only be distinguished by the intensity, the intensity histograms for the grayscale cars are shown in Fig. 9(a)-(d). Because the saturation reveals useful information to distinguish the colored cars, the grayscale cars, and the background, the saturation histograms for the colored cars are shown in Fig. 9(e)-(f). In these figures, the histogram of the background (red line) is averaged over several backgrounds of complex images. Moreover, the value of the blue and red lines in y-axes represent the ratio of the accumulated pixel number that belongs to each intensity (or saturation) value over the total pixel number of the car body or the background.

In Fig. 9(a), the intensity histogram of a black car has a peak region in the low intensity interval. Numerous pixels which belong to the car body and have considerably dark colors contribute to this peak. Therefore, a part of the car body can be included in the top three

candidates and selected as the seed region by our proposed seed selection strategy. However, because of the effect of light reflection, the intensity distribution of the car body is widely spread and considerably overlapped with the intensity distribution of the background. If a small number of color bins ($BinNum$) are assigned, a number of parts of the car body may be easily merged by the background. Therefore, a larger number of color bins ($BinNum$) is required. Under this condition, the car body will be divided into many small pieces. Hence, the number of macro merging times ($MergNum$) must be assigned a larger value to collect these small pieces and form up the whole car body.



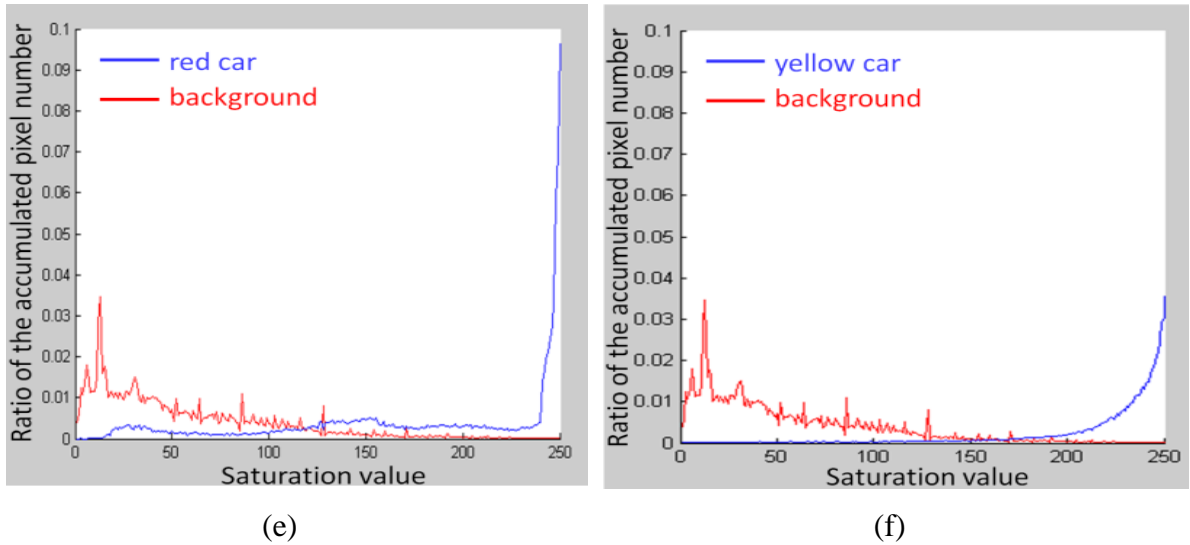


Fig. 9 The intensity (a)-(d) and saturation (e)-(f) histograms of six cars (blue lines) with black, white, dark-gray, light-gray, red and yellow colors and the background (red lines)

In Fig. 9(b), the intensity histogram of a white car has a peak region in the high intensity interval. Abundant pixels that belong to the car body with very bright color appear in the car image. Therefore, a part of the car body can be included in the top three candidates and selected as the seed region by our proposed seed selection strategy. The intensity distribution of the white car is not spread as widely as that of the black car indicating that the effect of light reflection on white cars does not split the car body into numerous small pieces. Therefore, a finite number (for example, 3~5) of the macro merging times ($MergNum$) is sufficient to form up the whole car body. However, the intensity distributions of the car body are overlapped by the distribution of the background. A larger number of color bins ($BinNum$) are still required to separate the car body and the background.

The intensity histogram of a dark-gray car and a light-gray car are displayed in Fig. 9(c) and (d). One or more peak regions are observed in these histograms. Under our seed region selection strategy, the region with either the minimal (in dark car category) or the maximal (in bright car category) intensity from the top-three largest regions will be selected as the seed region. When the region with minimal intensity is selected, some large parts which usually appear in the darker interval but belong to the background (for example, the shadow,

windshield, or bumper) may be selected as a wrong seed. While, if the region with maximal intensity is selected as the white car cases, a correct seed region which is selected from the car body can still be obtained.

In addition, the histogram of the background is less distributed in high intensity interval than in low intensity interval, because the effect of light reflection is not as obvious in the background as in the car body. Since only the top-three largest regions in the inner part are considered as the seed candidates, the small regions or the regions which extend to the outer part will not be selected as the seed region, even they have colors with high intensity. Moreover, light reflection usually changes the color of some car body parts to colors with higher intensity. Therefore, one of these parts which has the high intensity would belong to the top-three largest regions. Therefore, this region will be correctly selected as the seed region by the proposed seed selection strategy for bright car category.

If we create the fourth state which selects the seed region whose average intensity is mostly close to the middle of the histogram, some parts of the background such as asphalt road or cement wall may easily be selected as a wrong seed region. Furthermore, a bad car body will be determined. Increasing the number of the states does not improve the performance. Therefore, triple states for seed selection are adequate to extract the car body and further improve the color classification.

In some extreme cases where the intensity of a gray car is very dark, the seed selection strategy for bright car may not be suitable. No large parts of the car body with high intensity belong to the top-three largest regions. A wrong seed belonging to the background with middle intensity may be selected. However, the seed selection for dark car category is invoked simultaneously and it may select a more proper seed region. If the very dark and undesirable regions are not included in the top-three largest regions, a part of the car body can be correctly selected as the seed region. Even in the worse case, a considerably dark and undesirable region, such as the shadow, windshield, or bumper, is selected as the seed region.

Because these regions are neighboring to the car body and they have colors with similar intensity, the car body can be merged by the seed region and the whole car body can still be recovered; though some undesirable regions are also included. Therefore, triple states strategy for seed selection is sufficient to handle these extreme cases.

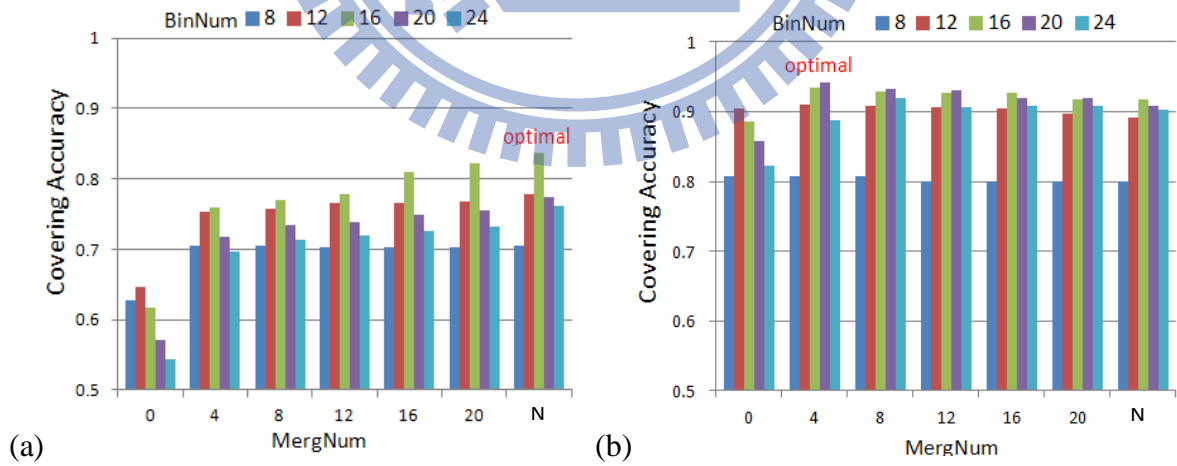
From Fig. 9(c) and (d), because the intensity distribution of the car body in dark-gray and light-gray cases is mostly overlapped with those of the backgrounds, a larger number of color bins (*BinNum*) are required. Moreover, a finite number of the macro merging times (*MergNum*) is sufficient to form up the whole car body because the essential parts of the intensity distribution are concentrated.

In the colored car cases as shown in Figs. 9(e) and (f), the saturation distributions of the red and the yellow cars are concentrated and distinguishable from the background after the specular-free processing. The proposed seed selection strategy for colored cars can extract the correct seed by comparing the saturation and selecting the largest area. To consider the efficiency of the system, a small number of color bins (*BinNum*) and a small number of macro merging times (*MergNum*) are sufficient for the red and yellow cars. In fact, all types of colored cars have concentrated saturation distributions. Therefore, the same processes and parameters are adopted. Single colored category is sufficient to cover all colored car cases.

Next, we analyze the performance in a quantitative manner. The covering accuracy (*CovAcc*), which is defined by Eq. (3), is considered as the performance index. The ground truth of the car body (CBR_{gt}) is manually extracted on each vehicle image. The covering accuracy computes the intersection of the extracted car body (CBR_{cat}) and the ground truth car body. The parameters which extract the car body with maximal covering accuracy are considered the optimal parameters.

$$CovAcc(CBR_{cat}) = \frac{\# \text{ of } px \in CBR_{cat} \cap CBR_{gt} + \# \text{ of } px \notin CBR_{cat} \cap CBR_{not_gt}}{\# \text{ of } px \in Img} \quad (3)$$

For the parameter *BinNum*, 8, 12, 16, 20, and 24 are tested, and for the parameter *MergNum*, 0, 4, 8, 12, 16, 20, and N (the number of segmented regions) are tested. The tri-states algorithm is executed by using these parameters on the car images that are stored in our collected datasets. The average covering accuracies on black, white, dark-gray, light-gray, and colored cars are shown in Fig. 10(a)-(e), respectively. The results with optimal parameters are marked in these figures. From Fig. 10(b)-(d), we can see that the optimal parameters in the dark-gray cars and the light-gray cars are identical to those in the white cars. As previous analysis, the seed region selection strategies for these three cases are also identical. To simplify the system architecture, we integrated the white, light-gray, and dark-gray cars into one group and termed as the bright car category. The dark car category covers the black cars and the extreme cases of dark-gray cars with very low intensity. All types of colored cars are integrated as the colored car category. Therefore, a tri-states (dark, bright, colored) architecture is proposed. Three states are separately executed with their (nearly) optimal parameters, listed in Fig. 10(f).



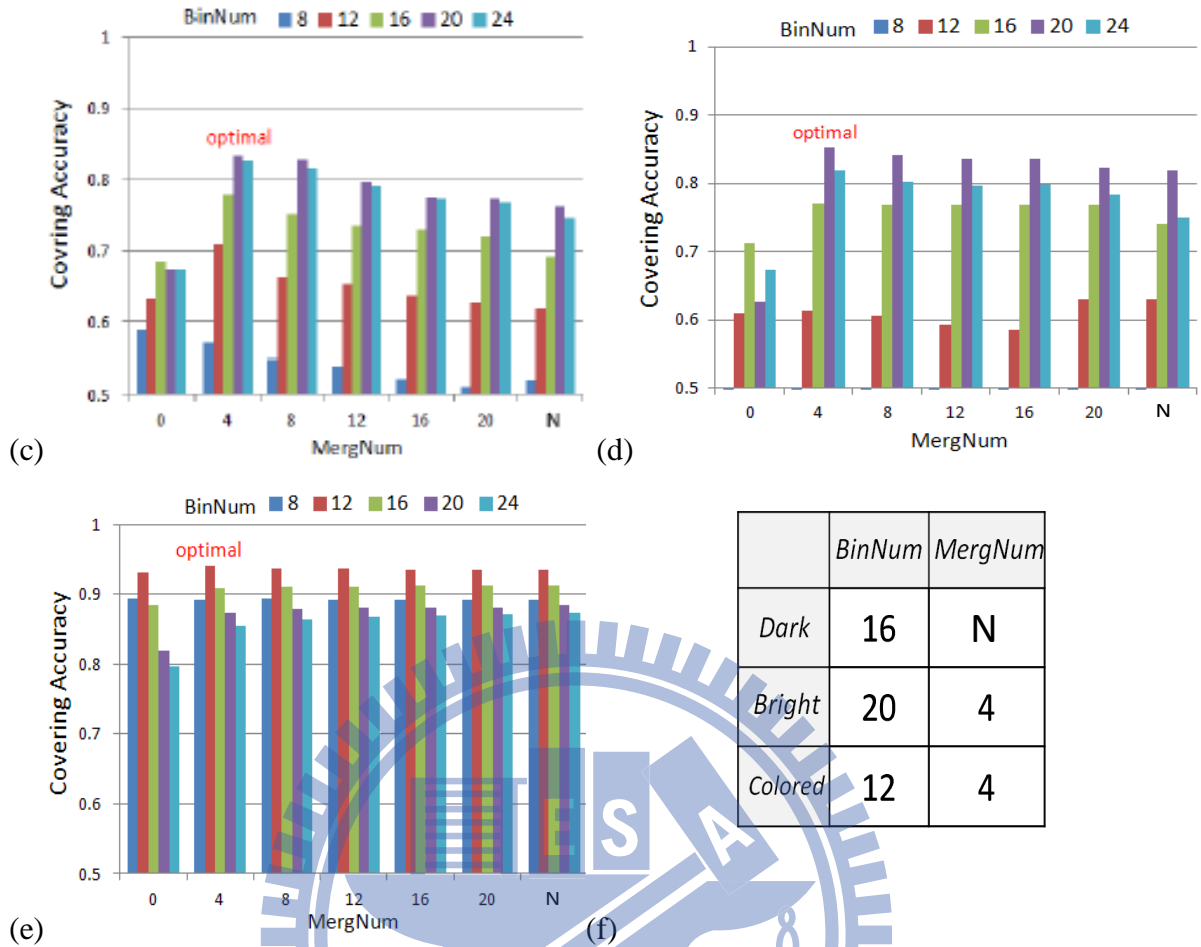


Fig. 10 The CovAcc analysis with various *BinNum* and *MergNum* parameters for (a) black, (b) white, (c) dark-gray, (d) light-gray, and (e) colored cars. (f) Optimal parameters table

3.3 Car body determination

A car body should be the most significant object in a vehicle image. If a region contains large number of pixels included in the inner part, the region is more significant. Eq. (4) shows a significance factor that is the number of pixels in the inner part minus the number of pixels in the outer part, and the result is divided by the area size of the inner part. If the candidate is not a car body, it may be split into many small pieces. Therefore, a compactness factor, defined in Eq. (5), is utilized to test the scatter degree of each CBR candidate. The scatter degree is related to the number of involved edge pixels of the CBR. The pixel is considered an edge pixel when it at least has a neighboring (7×7) pixel not belonging to the CBR. The

compactness factor is the ratio of non-edge pixels to all pixels in the CBR candidate. After normalizing the two factors within the range of [0, 1], they are multiplied together. The candidate with the highest score is selected to be the car body as Eq. (6).

$$Sign(CBR_i) = \frac{\# px \in CBR_i \cap InnerImg - \# px \in CBR_i \cap OuterImg}{\# px \in CBR_i \cap InnerImg}, \quad i \in \{bright, dark, colored\} \quad (4)$$

$$Comp(CBR_i) = \frac{\# px \notin edge(CBR_i)}{\# px \in CBR_i}, \quad i \in \{bright, dark, colored\} \quad (5)$$

$$CBR_{select} = \arg \max_{CBR_i} (Sign(CBR_i) \times Comp(CBR_i)) \quad i \in \{bright, dark, colored\} \quad (6)$$

Table 2 The generated CBRs for the white, orange, and black cars in Fig. 6(a)-(c). The significance and compactness factors for each generated CBR are shown below the CBR. For each car, the CBR with the red boundary is selected as the car body region

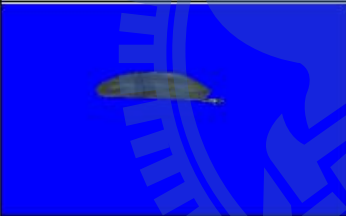

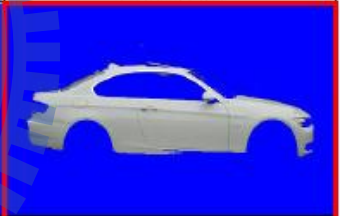
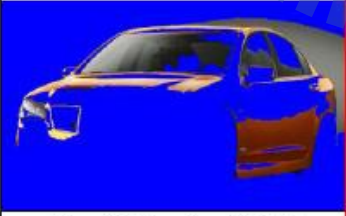

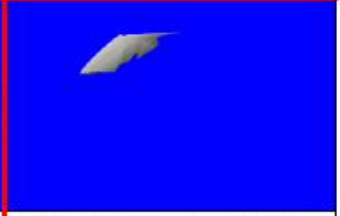


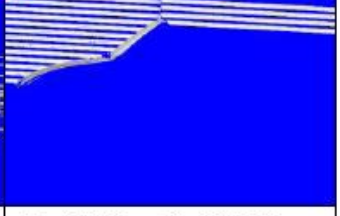
Category	CBR _{dark}	CBR _{colored}	CBR _{bright}
White car			
	Sign(CBR _{dark}) = 0.3396 Comp(CBR _{dark}) = 0.7227 Sign × Comp = 0.2452	Sign(CBR _{colored}) = 0.4325 Comp(CBR _{colored}) = 0.5794 Sign × Comp = 0.2506	Sign(CBR _{bright}) = 0.7042 Comp(CBR _{bright}) = 0.8052 Sign × Comp = 0.5670
Orange car			
	Sign(CBR _{dark}) = 0.5928 Comp(CBR _{dark}) = 0.6754 Sign × Comp = 0.4004	Sign(CBR _{colored}) = 0.9151 Comp(CBR _{colored}) = 0.8061 Sign × Comp = 0.7377	Sign(CBR _{bright}) = 0.2993 Comp(CBR _{bright}) = 0.7162 Sign × Comp = 0.2144
Black car			
	Sign(CBR _{dark}) = 0.7914 Comp(CBR _{dark}) = 0.8763 Sign × Comp = 0.6935	Sign(CBR _{colored}) = 0.7893 Comp(CBR _{colored}) = 0.6392 Sign × Comp = 0.5045	Sign(CBR _{bright}) = 0.0899 Comp(CBR _{bright}) = 0.2946 Sign × Comp = 0.0265

Table 2 shows the three CBR candidates, and their respective significance and compactness factors for the three cars shown in Fig. 11(a)-(c). For the white car, an improper dark seed from the windshield generates CBR_{dark} ; therefore, it has low significance. Because the specular-free operator is unsuitable for white cars, the compactness and significance of $CBR_{colored}$ are small. CBR_{bright} has good significance and compactness and is selected as the extracted car body. For the orange car, the dark seed in CBR_{dark} leads to low significance and low compactness. Since the specular-free operator is adopted, the significance and compactness of $CBR_{colored}$ is evidently higher to those in CBR_{bright} , enabling selection of $CBR_{colored}$. For the black car, the specular-free operator destroys the image, so the compactness of $CBR_{colored}$ is small. An improper seed generates CBR_{bright} , so that the significance and the compactness are poor. By having a proper seed and merging process, CBR_{dark} displays good significance and compactness and is selected as the car body.

3.4 Car color classification

Numerous vehicle color classification methods have been proposed in the past decade. HSV color model reduces the effect of circumstance luminance compared to RGB color space [51], and SVM shows excellent capability for identifying features [52, 53]. Although some studies claim a good successful rate, their performances are still influenced by impure background pixels and improper foreground pixels [11]. Since this study executed a beneficial car body extraction algorithm as described in the previous chapters, the color classification methods can achieve high performance by only considering the selected car body.

For each pixel in CBR, the RGB color value $(R_{CBR}, G_{CBR}, B_{CBR})$ is converted to the HSV color value $(H_{CBR}, S_{CBR}, V_{CBR})$ before computing the 3D HSV color histogram of the pixels in the CBR. Fig. 11 shows the hierarchical SVM classifier to identify the color type of the extracted car body. A binary SVM classifier: $SVM^2_{CG}(S_{CBR})$ is performed on the S_{CBR}

(Saturation) channel to distinguish the colored cars from the grayscale cars in the first layer. In the second layer, the more precise color types are going to be classified. If the category is colored, a multi-class SVM: $SVM^M_{colored}(H_{CBR})$ is performed on the H_{CBR} (Hue) channel to classify the car body into one of the colored types: red, orange, yellow, green, blue, purple, and pink. If the category is grayscale, the other multi-class SVM: $SVM^M_{WSB}(V_{CBR})$ is performed on the V_{CBR} (Value) channel to classify the car body into one of the grayscale types: white, gray, and black.

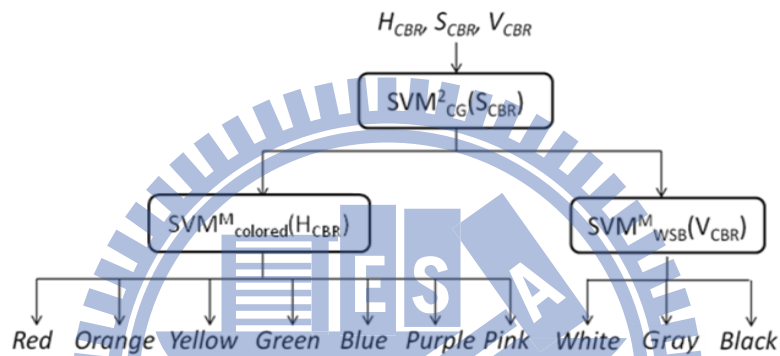


Fig. 11 Hierarchical SVM of color type classification

3.5 Experiments of car color determination

The experiments were performed and compared to demonstrate the superiority of our proposed ideas on tri-states car body extraction and color classification. In Section 3.5.1, several car body extraction algorithms are introduced for comparison. Section 3.5.2 describes two car image databases. In Section 3.5.3, four metrics are described for evaluating the performance of the tri-states method. The efficiency of each step of the tri-states method is also demonstrated in this section.

3.5.1 Experimental description

Two current car body extraction and color classification methods were implemented for comparison to reveal the advantages of the proposed tri-states algorithm.

































- 1) Homogeneous SVM (homo-SVM) method: After the image is segmented into multiple regions, the largest part in the inner part, in which all its pixels have similar colors, is selected. This part is subsequently fed into the SVM [7] to classify its color. Through the process for parameter selection, the number of color bins (*BinNum*) used in this method is selected as 12.
- 2) Removal rule method [11]: After the image is segmented into multiple regions, the dominant color of the central part is computed. The segmented regions are subsequently excluded with average colors that differ from the dominant color, and the regions with an excessively large or small average intensity are removed. Finally, the remaining regions are grouped as the car body and the color is classified based only on the pixels within the car body. Through the process for parameter selection, the number of color bins (*BinNum*) used in this method is selected as 16.
- 3) Tri-states method: The proposed method of this study. The procedure of the tri-states framework is performed as shown in Fig. 5. This method provides distinct strategies for selecting seeds from bright, dark, and colored cars. In addition, the seed region is not confined to similar color to the dominant color of the central part of the image. Anti-reflection design (SARM algorithm) is also provided to enhance the performance of car body extraction and color classification. Through the process of parameter selection, the number of color bins (*BinNum*) and the number of macro merging times (*MergNum*) are selected according to the results decided in Chapter 3.1.5.

3.5.2 Collected dataset

To test the proposed tri-states method and the two compared methods, two vehicle image datasets were collected to stimulate intelligent transportation system (ITS) and content-based image retrieval (CBIR) scenarios. The ITS dataset captures car images from real traffic streams obtained using cameras. The camera is placed at a fixed position; hence the view of

the captured vehicle is known. Because of the static background, a background subtraction technique [21] is adopted. The vehicle images in this dataset include only the interesting foreground regions of the vehicles. Seven common colors are present in the dataset: white, black, gray, red, yellow, green, and blue. Each color set contains 100 car images. Table 3 displays several sample images from this dataset.

Table 3 Sample images from the ITS dataset

Color	ITS				
Black					
Gray					
White					
Red					
Yellow					
Green					
Blue					

The CBIR dataset collects car images from the Internet. Several sample images of these two datasets are listed in Table 4. Since vehicle images on the Internet may be captured from any viewpoint and the background model cannot be constructed, vehicle color classification in

this scenario is more challenging. Not only the seven color type cars exist as in ITS dataset, but the cars with three rare color types: pink, purple, and orange, are also collected in the CBIR dataset. Each color type contains about 30 car images, and the total number of images is over 310.

Table 4 Sample images from the CBIR-simple and CBIR-complex dataset

	CBIR-simple	CBIR-complex
Black		
Gray		
White		
Red		
Orange		
Yellow		
Green		
Purple		
Blue		
Pink		

We further divide these images into two equivalent quantity subsets. One subset contains the images with more complicated backgrounds, such as complicated building. Furthermore, car bodies may not be certainly located at the central part, or not guarantee to occupy a sufficient large partition of the image. We term this dataset as the CBIR-complex dataset. The rest images are collected as the CBIR-simple dataset.

In the experiments, the ten-fold cross validation is adopted to evaluate the performances. In other words, each dataset of ITS, CBIR-simple, and CBIR-complex is divided into ten subsamples. When each dataset is tested, the operation of using one subsample for testing and the other nine subsamples for training is repeated for ten times. The successful rates for color classification are averaged over all the subsamples. The datasets are available at [54].

3.5.3 Experimental results

3.5.3.1 Covering ratio performance

Table 5 shows the confusion matrices of color category classification of the proposed tri-states method performing on the ITS, CBIR-simple and CBIR-complex datasets. In the ITS dataset, the car images with categories of 230 bright, 170 dark and 400 colored are tested. Only two bright cars are misclassified into the colored category, and all dark and colored cars are correctly classified into their categories. The average accuracy of color category classification on the ITS dataset is 99.7 %. In the CBIR-simple dataset, 30 bright, 20 dark, and 105 colored car images are classified. Only 1 colored car is misclassified by the tri-states method. The average accuracy of color category classification on the CBIR-simple dataset is 99.3 %. In the CBIR-complex dataset, 30 bright, 20 dark and 105 colored car images are classified their categories. Only 1 dark car and 3 colored cars are misclassified. The average accuracy of color category classification on the CBIR-complex dataset is 97.4 %.

Table 5 Confusion matrices of color category classification on ITS, CBIR-simple and CBIR-complex datasets

	ITS				CBIR-simple				CBIR-complex			
T/D	Bright	Dark	Colored	AC _{cat}	Bright	Dark	Colored	AC _{cat}	Bright	Dark	Colored	AC _{cat}
Bright	228	0	2	99.1%	30	0	0	100.0%	30	0	0	100.0%
Dark	0	170	0	100.0%	0	20	0	100.0%	1	19	0	95.0%
Colored	0	0	300	100.0%	1	0	104	99.1%	3	0	102	97.1%

3.5.3.2 Covering ratio performance

Covering ratio which is defined as the percentage of pixels covering both the extracted car body CBR_{cat} and the ground truth car body CBR_{gt} can be taken as the performance index. The covering ratio comprises the covering precision ($CovPrec$) and the covering recall ($CovRec$), as defined by Eq. (7)-(8). The covering precision shows the percentage of the pixels in the extracted car body belongs to the real car body. And, the covering recall shows the percentage of the pixels in the real car body has been extracted.

$$CovPrec(CBR_{cat}) = \frac{\# \text{ of } px \in CBR_{cat} \cap CBR_{gt}}{\# \text{ of } px \in CBR_{cat}} \quad (7)$$

$$CovRec(CBR_{cat}) = \frac{\# \text{ of } px \in CBR_{cat} \cap CBR_{gt}}{\# \text{ of } px \in CBR_{gt}} \quad (8)$$

Fig. 12 shows the covering precisions and the covering recalls of the homo-SVM method, the removal rule method, and the tri-states method on the ITS, CBIR-simple, and CBIR-complex datasets, respectively. In Fig. 12(a), the covering precisions of the three methods are tested on the ITS dataset. Because most of the background pixels are subtracted in this scenario, the car body can be detected more easily so that all the methods provide satisfactory covering precisions. The average covering precisions of the homo-SVM, removal rule method, and tri-states approaches are 81%, 78% and, 87%, respectively. The tri-states method is slightly superior to the other two methods. In Fig. 12(d), the average covering

recalls of the three methods are 56%, 71% and, 88%, respectively. The average covering recall of the homo-SVM is inferior to that of the other approaches because only the most significant part is selected. The tri-states method has merging process with optimal parameters for various color car category, the average covering recall of the tri-states method is obviously superior to the other approaches.

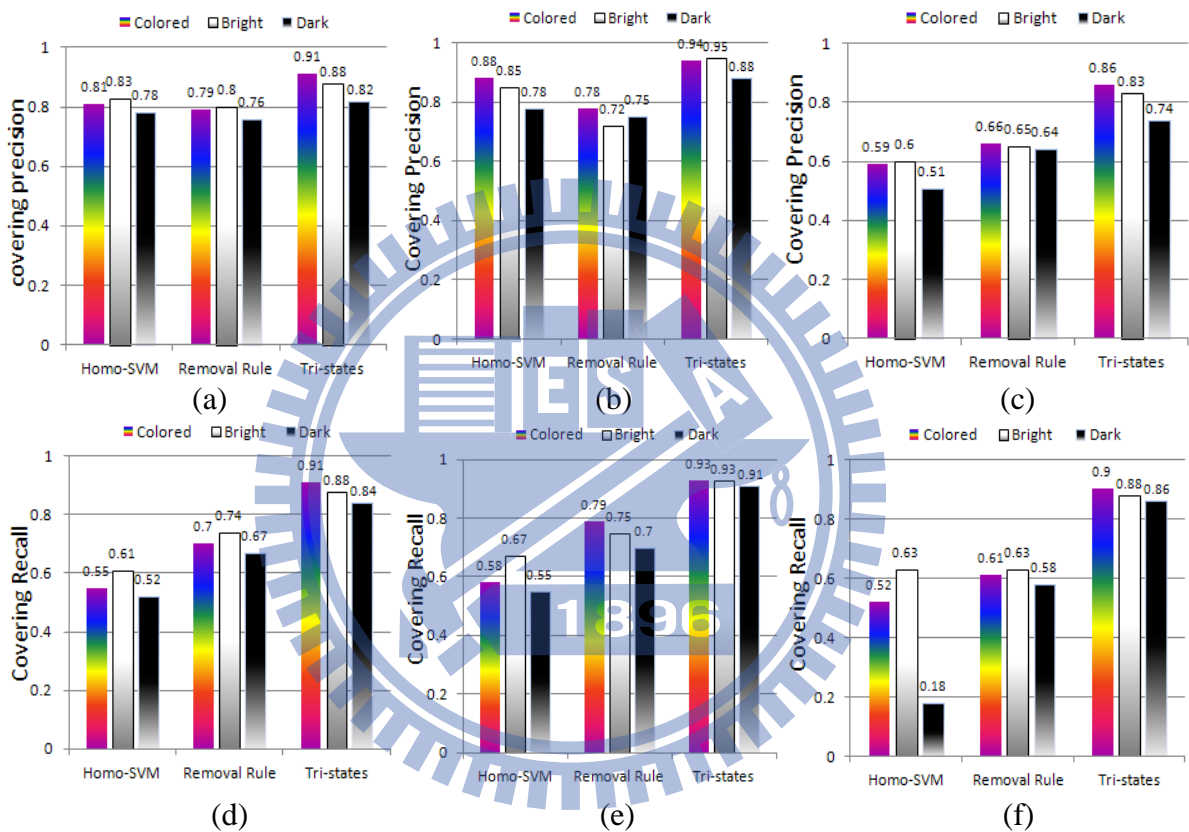


Fig. 12 Covering precisions of three compared methods on (a) ITS, (b) CBIR-simple, and (c) CBIR-complex datasets for each color category. Covering recalls of three compared methods on (d) ITS, (e) CBIR-simple, and (f) CBIR-complex datasets for each color category

In Fig. 12(b) and 12(e), the three compared methods are tested on the CBIR-simple dataset. The average covering precisions of these methods are 81%, 75% and, 92%, respectively. The average covering recalls of the three methods are 60%, 75% and, 92%, respectively. The performances of the methods in the CBIR dataset are similar to those in the ITS dataset. The tri-states method provides superior covering recall to the other two methods.

Fig. 12(c) shows the covering precisions of the three compared methods on the CBIR-complex dataset. In this scenario, the multi-colored background is included in the image; therefore, the regions with the same color to the vehicle are easily falsely distinguished as the car body by the homo-SVM method. Hence, the average covering precisions of the homo-SVM method decrease to 57%. In addition, the car body in this scenario may not completely appear in the central part of the image. Therefore, the average covering precisions of the removal rule method only achieve 65%. The average covering precisions of the tri-states method are 81%. Although the covering precision of the tri-states method in the CBIR-complex scenario is slightly lower than those in the ITS and CBIR-simple scenarios, its superiority to the other methods is very obvious. In Fig. 12(f), we see that the covering recalls of the homo-SVM method and the removal rule method are only 45% and 61%, respectively. However, the average covering recall of the tri-states method can still achieve 88%, even in this CRIR-complex scenario. We conclude that the tri-states method can provide significantly superior covering precision and covering recall to the homo-SVM method and the removal rule method on the ITS, CBIR-simple, and CBIR-complex scenarios.

3.5.3.3 Color deviation

Color deviation (*ColDev*) shows the average color difference between the extracted car body CBR_{cat} and the ground truth car body CBR_{gt} . Manually select k pixels spreading in the CBR_{cat} to attain the hue ($H^1_{gt} \dots H^k_{gt}$) and intensity ($I^1_{gt} \dots I^k_{gt}$) values of these pixels as the ground truth colors of the car. For colored cars, the color deviation is computed by the average difference between the hue values of the pixels (px) in the CBR_{cat} and the ground truth hue, as shown in Eq. (10). For bright or dark cars, color deviation (*ColDev*) is computed by the average difference between the intensity values of the pixels in the CBR_{cat} and those of the ground truth intensity, as shown in Eq. (11).

$$ColDev(CBR_{cat}) = \begin{cases} HueDev(CBR_{cat}), & \text{if } cat \in \text{colored} \\ IntDev(CBR_{cat}), & \text{if } cat \in \text{bright or dark} \end{cases} \quad (9)$$

$$HueDev(CBR_{cat}) = \frac{\sum_{px \in CBR_{cat}} \min_{1 \leq i \leq k} (|Hue(px) - H_{gt}^i|)}{\# px \in CBR_{cat}} \quad (10)$$

$$IntDev(CBR_{cat}) = \frac{\sum_{px \in CBR_{cat}} \min_{1 \leq i \leq k} (|Int(px) - I_{gt}^i|)}{\# px \in CBR_{cat}} \quad (11)$$

The extracted car body may involve numerous pixels that do not belong to the actual car body. The color histogram may be deviated by these pixels. The color deviation, as defined in Eqs. (9)-(11), represent the purity of color in the extracted car body. The hue deviations of the car bodies that are extracted from the three methods are computed for the colored car cases on the ITS, CBIR-simple and CBIR-complex datasets. The intensity deviations are also computed for the grayscale car cases. The accumulated percentage of these car bodies for colored and grayscale cars are shown in Fig. 13. Fig. 13(a) shows the hue deviation when the ITS dataset are tested for the three compared methods. If the hue deviation smaller than 30 is tolerable, the accumulated percentages of hue deviation by the homo-SVM method, the removal rule method, and the tri-states method are 85%, 82%, and 96%, respectively. When the tolerance of the hue deviation increases to 60, the accumulated percentages by these three methods are 91%, 93%, and 98%, respectively. The performance of the homo-SVM method is superior to that of the removal rule method when more colors are required to be classified. If only a few types of colors, such as four colors: red, blue, yellow and green, to be required for classification, the removal rule method is superior to the homo-SVM method. The tri-states method extracts purer colors than the homo-SVM method and the removal rule method so that the tri-states method has better performance on hue deviation.

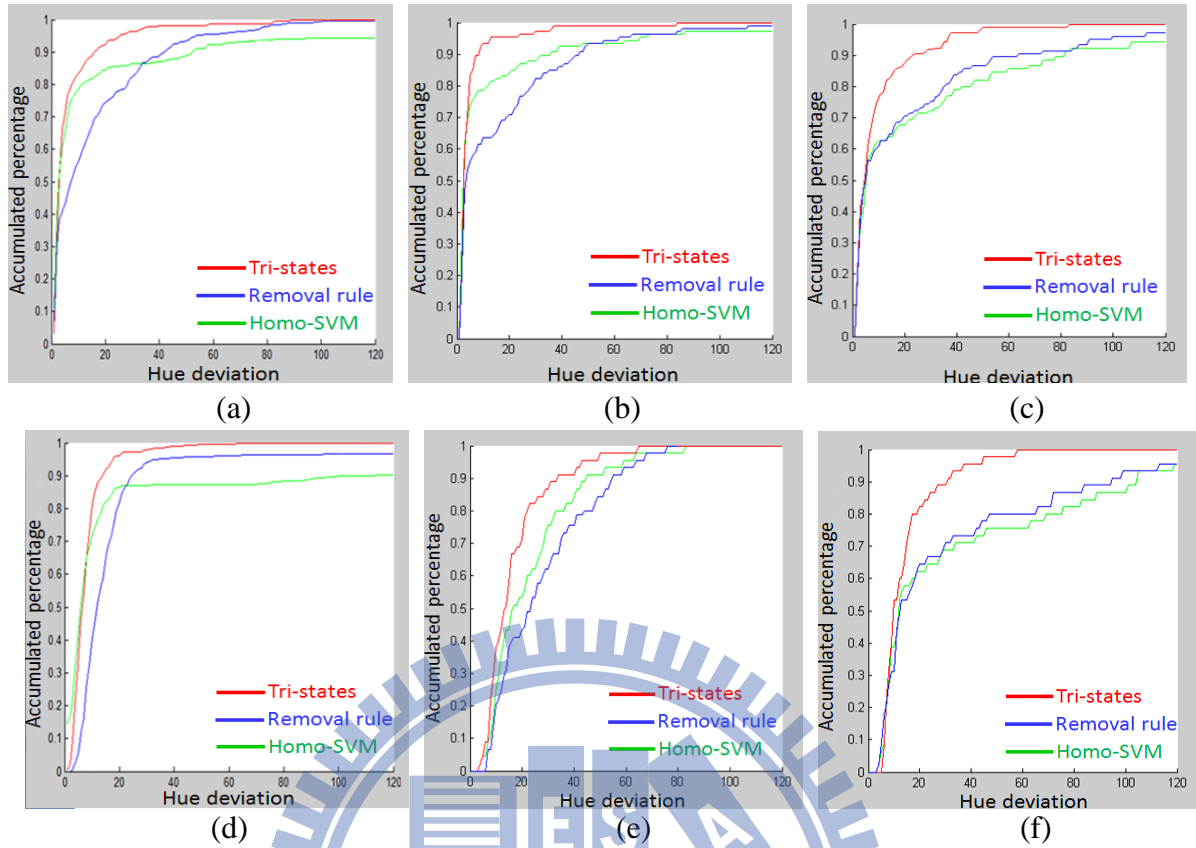


Fig. 13 Accumulated percentage of hue deviation by three compared methods on colored car images in (a) ITS, (b) CBIR-simple and (c) CBIR-complex datasets. Accumulated percentage of intensity deviation by three compared methods on grayscale car images in (d) ITS, (e) CBIR-simple and (f) CBIR-complex datasets

Fig. 13(b) shows the hue deviation when the CBIR-simple dataset are tested for the three compared methods. If the hue deviation smaller than 30 is tolerable, the accumulated percentages of hue deviation by the homo-SVM method, the removal rule method, and the tri-states method are 86%, 82%, and 96%, respectively. When the tolerance of the hue deviation increases to 60, the accumulated percentages by these three methods are 92%, 93%, and 99%, respectively. Because the removal rule method may obtain an incorrect dominant color due to the various viewpoints in CBIR scenario, the accumulated curve increases slower than the curve in the ITS scenario. The homo-SVM extracts purer color car body than the removal rule method because the background is more easily distinguished. The tri-states method selects more proper seeds than the other two methods and it can work well without

the limitation of viewpoint. Hence, the improvement of the tri-states method to the other approaches is notable in the CBIR-simple dataset.

Fig. 13(c) shows the hue deviation when the images in the CBIR-complex dataset are tested for the three compared methods. If the hue deviations smaller than 30 is tolerable, the accumulated percentages of hue deviation by the homo-SVM method, the removal rule method, and the tri-states method are 74%, 78%, and 90%, respectively. When the tolerance of the hue deviation increases to 60, the accumulated percentages of hue deviation by these three methods are 85%, 89%, and 99%, respectively. Because of the interference of the complicated background, much more incorrect seed region are selected by the removal rule method and the homo-SVM method. And since the tri-states method generates the triple car body candidates by proper strategies with (nearly) optimal parameters for various color car category, the car body can be extracted more accurately, thereby purer colors which approximate the real colors of tested cars are obtained. The superiority of the tri-states method to the other two compared methods is very obvious.

The grayscale cars are also tested on ITS, CBIR-simple, and CBIR-complex datasets and their intensity deviations are shown in Fig. 13 (d)-(f), respectively. In Fig. 13 (d), excluding the considerably small intensity deviation cases, the tri-states method is consistently superior to the other approaches. The removal rule method is superior to the homo-SVM method, when the tolerable intensity deviation is larger than 15. While in Fig. 13 (e), the removal rule is worse than the other methods due to the effect of viewpoint varying. In Fig. 13 (f) the homo-SVM method and the removal rule method are competitive and both of them obtain poor performance. The tri-states method is consistently superior to these two compared methods. By comparing Fig. 13 (c) and Fig. 13 (f), the superiorities of the tri-states method to the two compared methods are even more significantly in the grayscale car cases than in the colored car cases when they work on the CBIR-complex dataset.

3.5.3.4 Color classification accuracy

The accuracies of color category (AC_{cat}) and color type ($AC_{colorType}$) demonstrate the performance of color classification. For each image in a dataset, the color category and the color type are decided using several support vector machines (SVM). Two binary SVM: SVM^2_{CG} and SVM^2_{BD} compose a function SVM_{cat} to determine the color category of the extracted CBR_{cat} , as shown in Eq. (12). The SVM: SVM^2_{CG} distinguish colored cars from grayscale cars by saturation histogram. The SVM: SVM^2_{BD} separates bright and dark cars by intensity histogram. A color type decision function $SVM_{colorType}$ comprises two multi-class SVM: $SVM^M_{colored}$ and SVM^M_{WSB} , to determine the color type of the extracted CBR, as shown in Eq. (13). The SVM: $SVM^M_{colored}$ identifies multiple colors from the colored category. The SVM: SVM^M_{WSB} , further recognizes white, gray and black cars from the grayscale vehicles.

$$AC_{cat} = \frac{\sum_{i=1}^{\# \text{ of images}} SVM_{cat}(CBR_i) == SVM_{cat}(CBR_i^{gt})}{\# \text{ of images in the dataset}} \quad (12)$$

$$AC_{ColorType} = \frac{\sum_{i=1}^{\# \text{ of images}} SVM_{ColorType}(CBR_i) == SVM_{ColorType}(CBR_i^{gt})}{\# \text{ of images in the dataset}} \quad (13)$$

After the car body region has been extracted, the color histogram of the car body is computed and then the SVM function depicted in Fig. 11 is utilized to classify the color type. Fig. 14(a)-(c) show the accuracies of the color type classification on the extracted CBR by the three compared methods on ITS, CBIR-simple and CBIR-complex datasets, respectively.

In Fig. 14(a), the average accuracies of color type classification by the homo-SVM method, the removal rule method, and the tri-states, method on the ITS dataset are 91%, 94%, and 97%, respectively. For the colored cars in ITS scenario, because only four kinds of colors: red, blue, green and yellow, are needed to be classified, the hue deviation tolerance can be large. For the grayscale images, only three kinds of colors: black, gray and white, are needed

to be classified, the intensity deviation can also be large. All methods perform well under a large tolerance, but the tri-states method still provides approximately 6% and 3% improvements of classification accuracy compared with the homo-SVM method and the removal rule method, respectively.

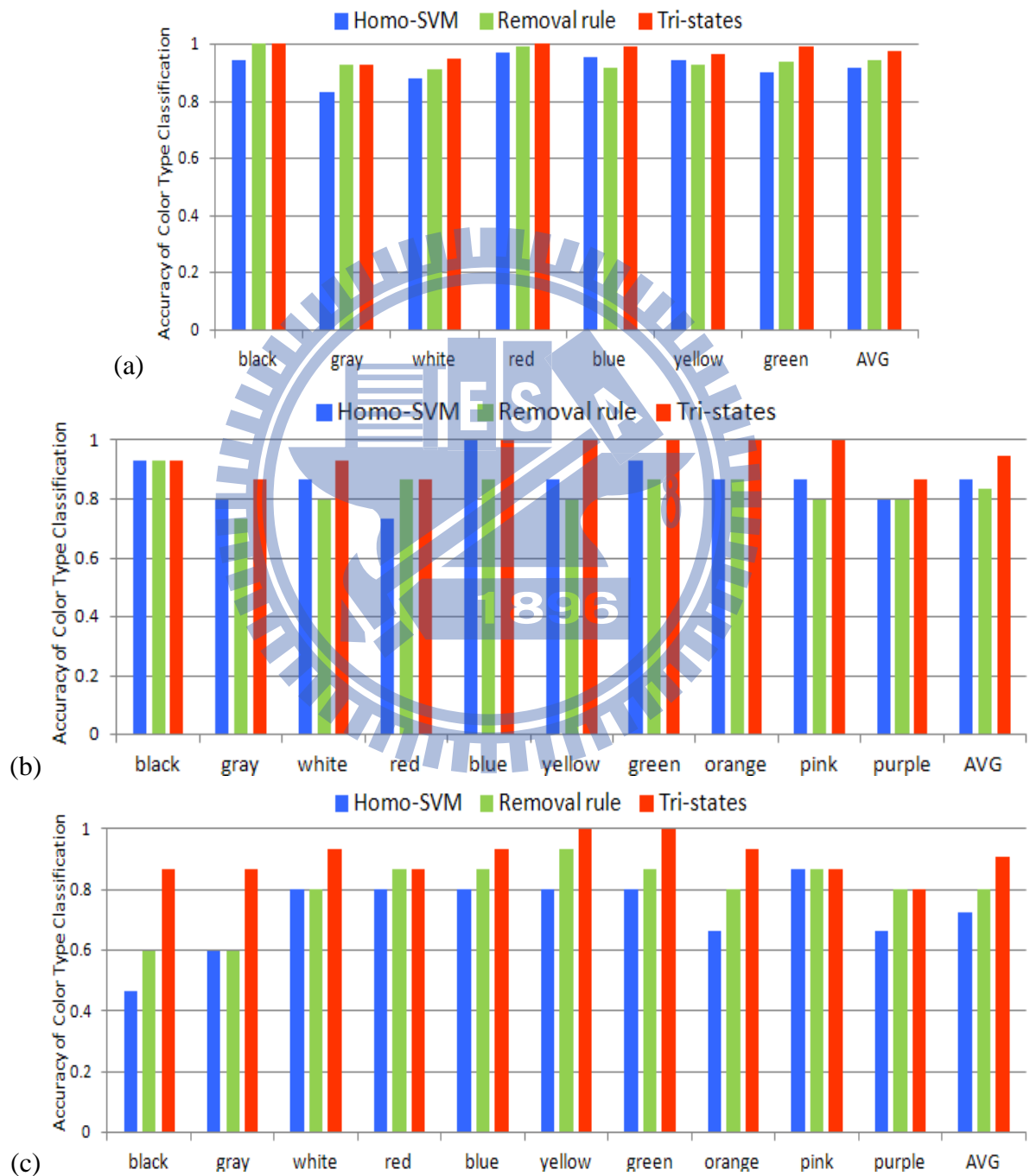


Fig. 14 Accuracies of color type classification of the three compared methods on in (a) ITS, (b) CBIR-simple and (c) CBIR-complex datasets

In Fig. 14(b), there are ten color types to be classified in the CBIR-simple dataset; hence, the tolerances of hue deviation should be smaller. The accuracies of color type classification will get lower and the removal rule method works the worst. The average accuracy of color type classification by the tri-states method is 94%. The superiorities of the proposed tri-states method to the homo-SVM method and the removal rule method are more significant and they are over 8% and 11%, respectively.

In Fig. 14(c), ten color type cars are classified in the CBIR-complex dataset. Due to the interference of complicated backgrounds, all methods in this dataset perform obviously worse than those in the other datasets. However, the average accuracy of color type classification by the tri-states method still achieve 91% and the average improvements of the tri-states method to the homo-SVM method and the removal rule method are over 18% and 10%, respectively.

3.5.3.5 Computation load

The average computation time of each component on an image with 100×100 pixels is reported in Table 6. The entire computation time in the proposed algorithm is approximately 0.057 sec per image. This length of computation time is applicable to most real-time systems for car body extraction and color classification. The computation time of the two compared algorithms: the homo-SVM and the removal rule methods are 0.015 sec and 0.021 sec, respectively. Although the tri-states method requires slightly more computation time than the two compared methods, it provides a more intact car body and higher color classification accuracy than the two compared methods.

Table 6 Average computation time of each component in the proposed car color determination algorithm and in the compared methods

<i>Method</i>	<i>Image segmentation</i>	<i>Candidates generation</i>	<i>Car body determination</i>	<i>Color type classification</i>	<i>Whole algorithm</i>
<i>Homo-SVM</i>	12 ms	NULL	1.4 ms	1.6 ms	15 ms
<i>Removal rule</i>	18 ms	NULL	1.4 ms	1.6 ms	21 ms
<i>Tri-states</i>	colored: 12 ms bright: 23ms dark: 18ms	colored: 0.28 ms bright: 0.28 ms dark : 0.42 ms	1.8 ms	1.6 ms	57.38 ms



Chapter 4 Car Pose Estimation Algorithm

This chapter addresses a proposed pose algorithm to compute the rotation angle of a tested car. The algorithm provides a sufficiently accurate pose for classifying size and recognizing models and it is rapid enough to generate initial pose for more complex pose estimation algorithm. The symmetric center on a vehicle image is searched and the distance between the symmetric center and the left (or right) boundary of the car is computed. The ratio of the distance to the car width is calculated and used to estimate the pose. The definition of the ratio and the way to search the symmetric center are introduced in Chapters 4.1 and 4.2. The experimental results are provided in Chapter 4.3.

4.1 Pose ratio computing

Generally, the left, right, top, and bottom ends (pos_{left} , pos_{right} , pos_{top} , pos_{bot}) of the bounding rectangle of a car can be extracted by frame difference [31] or background subtraction [33], as shown in Fig. 15(a). Conventionally, the geometric center generated by the four ends is deemed as the reference position for matching. The ratio of the car width ($wid = pos_{right} - pos_{left}$) to car height ($hgt = pos_{bot} - pos_{top}$) denoted by γ_{cov} is obtained as Eq. (14).

$$\gamma_{cov} = \frac{wid}{hgt} = \frac{pos_{right} - pos_{left}}{pos_{bot} - pos_{top}} \quad (14)$$

All orientations (pose angles) can be divided into four quadrants: (I: $0^\circ \sim 90^\circ$, II: $90^\circ \sim 180^\circ$, III: $180^\circ \sim 270^\circ$, and IV: $270^\circ \sim 360^\circ$). For each quadrant, we collected a set of standard templates with known pose angles (θ) and manually extracted the ratios on these templates. A monotonic function $Ratio_{cov}$ (where $\gamma_{cov} = Ratio_{cov}(\theta)$) can be constructed by these ratios for each quadrant, as shown in Fig. 15(b). To estimate the pose angle θ_{cov} ($\theta = \theta_{cov}$) of the tested car in a quadrant, the width and height of the car are extracted and the ratio

γ_{cov} is computed. Then, the pose angle of the tested car can be computed using the inverse function of $Ratio_{cov}$ as Eq. (15):

$$\theta_{cov} = Ratio_{cov}^{-1}(\gamma_{cov}) \quad (15)$$

Since various car models have different lengths and widths, each target car model has a unique function $Ratio_{cov}$ to estimate the pose in a quadrant. The ratio belonging to the target car model is used when determining whether a tested car belongs to that target car model. To extend the processing into multi-model recognition, as when identifying N target models, N $Ratio_{cov}$ functions are required to be trained in each quadrant. An estimated pose angle is produced by each function. The proposed system processes all target car models sequentially.

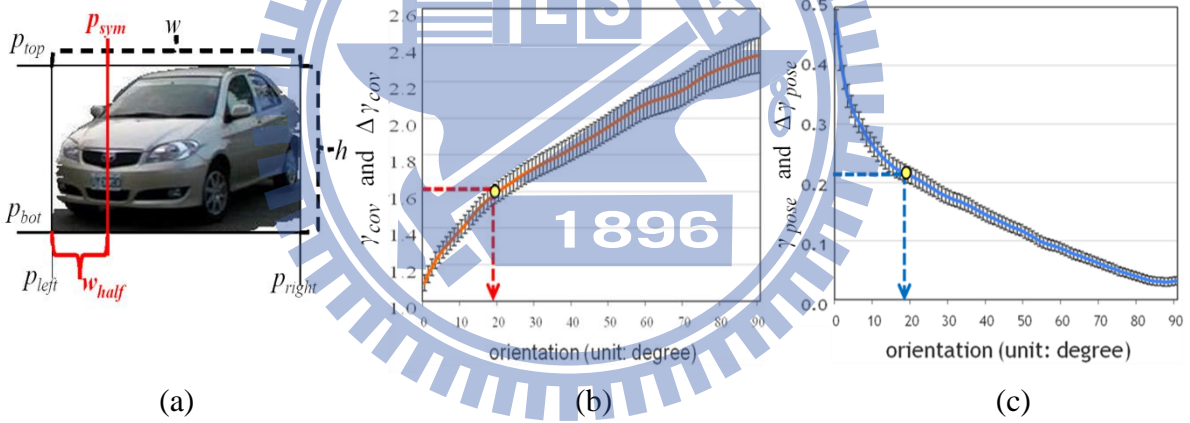


Fig. 15 (a) The left, right, top, bottom ends and the symmetric center of a car, (b) the ratio γ_{cov} of car width to car height (red curve) and its estimation error $\Delta\gamma_{cov}$ (error bar), (c) the ratio γ_{pos} of half width to car width (blue curve) and its estimation error $\Delta\gamma_{pos}$ (error bar)

For each target car model, if no prior information concerning the quadrant is available, the computed ratio γ_{cov} of the tested car image corresponds to four possible pose angles in the four quadrants. To determine which quadrant is the correct one, four template images of the target car model at the four pose angles are chosen. The tested car image is matched against the four template images, and the template image with the minimum matching error is

selected. The quadrant and pose angle of the selected template image are considered those of the tested car.

Because the bottom ends are easily interfered by some environmental effects, such as shadows, the ratio of the width to height may also be erroneous if a small tilt rotation between the camera and the vehicle exists. Hence, this approach generally cannot provide accurate enough pose angle. We searched for further information to improve pose estimation accuracy. The symmetric center pos_{cen} in the front (or rear) region of a car shown in Fig. 15(a) could be a good choice. The symmetric center can be viewed as a reference point for matching the car, and the half width wid_{half} of the front (or rear) region can be utilized to estimate the pose angle. When the car is orientated within quadrant I or III, the symmetric center is searched beginning from left toward right and the half width can be defined as $wid_{half} = pos_{cen} - pos_{left}$. If the car is orientated within quadrant II or IV, the algorithm in searching the symmetric center should begin from right toward left and the half width is defined as $wid_{half} = pos_{right} - pos_{cen}$.

A new ratio of wid_{half} to wid , denoted by γ_{pose} , is proposed in Eq. (16) for estimating the pose angle of the tested car. For each target car model, a set of standard templates were collected with known pose angles (θ) and the ratios γ_{pose} of the templates were manually extracted. In each quadrant, a monotonic function $Ratio_{pose}$ (where $\gamma_{pose} = Ratio_{pose}(\theta)$) can be constructed by the template ratios, as shown in Fig. 15(c). Each target model obtains its unique function $Ratio_{pose}$ in a quadrant. To estimate the pose angle θ_{pose} ($\theta = \theta_{pose}$) of the tested car, the ratio γ_{pose} is first obtained, whereupon the pose angle of the tested car can be computed by the inverse function of $Ratio_{pose}$ as Eq. (17). In multi-model recognition, as when identifying N target car models, N $Ratio_{pose}$ functions are required to be trained in each quadrant. An estimated pose angle is produced by each function, and the proposed system processes all target car models sequentially.

$$\gamma_{pose} = \frac{wid_{half}}{wid} = \begin{cases} \frac{pos_{cen} - pos_{left}}{pos_{right} - pos_{left}}, & \text{if } \theta \in [0^\circ, 90^\circ] \text{ or } [180^\circ, 270^\circ] \\ \frac{pos_{right} - pos_{cen}}{pos_{right} - pos_{left}}, & \text{if } \theta \in [90^\circ, 180^\circ] \text{ or } [270^\circ, 360^\circ] \end{cases} \quad (16)$$

$$\theta_{pose} = Ratio_{pose}^{-1}(\gamma_{pose}) \quad (17)$$

If no prior information about the quadrant is available, a symmetric center candidate on the left side and another candidate on the right side are sought simultaneously. Two half widths ($wid_{half} = pos_{cen} - pos_{left}$ or $pos_{right} - pos_{cen}$) and two ratios γ_{pose} are computed. Each ratio corresponds to two pose angles in two quadrants: (I and III), or (II and IV), so two ratios produce four possible pose angles in the four quadrants for each target car model. The template images belonging to each target car model depicting the four possible poses by γ_{pose} are acquired. Similar to the processing of γ_{cov} , the tested car image is matched with the template images in the pose and center estimation step, and the template image with the minimum matching error is selected. The quadrant and pose angle of the selected template image are adopted as those of the tested car for each target car model.

4.2 Symmetric center detection

To estimate the pose angle, the symmetric center has to be obtained. Traditional symmetry-based methods [40-44] were developed to acquire the symmetric centers of cars within a narrow range of orientations. To accommodate wider range of poses, a more robust algorithm to identify the symmetric center is proposed. Fig. 16 shows an example to introduce the algorithm.

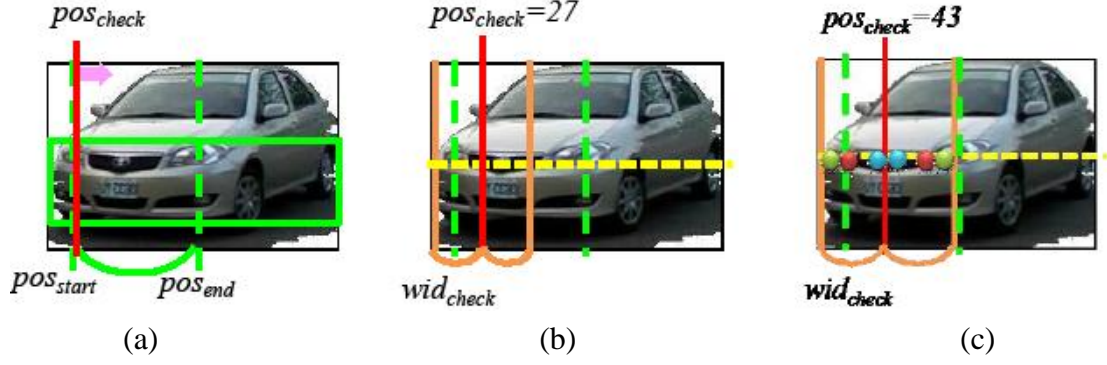


Fig. 16 (a) the planar area and the possible range to search for the symmetric center, (b) the check width and the check point at x -axis = 27, (c) the check point at 43 and the pixels (red to red, blue to blue, green to green) in correspondence to compute ASD

First, we select an area on about the same plane as the green rectangle in Fig. 16(a) and check the symmetric property of the pixels in this area. Since the pose angle can be coarsely estimated by the four ends of the car, a possible range $[pos_{start}, pos_{end}]$ which the symmetric center may appear is assigned. In Fig. 16(b), for each row in the selected area, a check point (pos_{check}) moves gradually from pos_{start} to pos_{end} and the check width ($wid_{check} = pos_{check} - pos_{left}$) increases. For each check point, the accumulated symmetric difference (ASD) is computed as follows.

$$ASD(pos_{check}) = \begin{cases} \sum_{x=pos_{left}}^{x=pos_{check}} \frac{|Int(2 \times pos_{check} - x, y) - Int(x, y)|}{2 \times (pos_{check} - pos_{left})}, & \text{if } \theta \in [0^\circ, 90^\circ] \text{ or } [180^\circ, 270^\circ] \\ \sum_{x=pos_{right}}^{x=pos_{check}} \frac{|Int(2 \times pos_{check} - x, y) - Int(x, y)|}{2 \times (pos_{right} - pos_{check})}, & \text{if } \theta \in [90^\circ, 180^\circ] \text{ or } [270^\circ, 360^\circ] \end{cases} \quad (18)$$

, where $Int(x,y)$ is the intensity value at the pixel (x,y) on a tested car image, and $pos_{check} \in [pos_{start}, pos_{end}]$

In Fig. 16(c), the red pixel in the left side of pos_{check} corresponds to the red pixel in the right side. In the same way, the same color pixels in the left side and the right side of pos_{check} are in correspondence. The intensity differences of the corresponding pixels are accumulated to acquire the ASD. The blue curve in Fig. 17 shows the ASD results over all check points in

the check range of Fig. 16. Because the ASD of the symmetric center should be relatively lower than those of neighboring pixels, a valley would appear in the ASD distribution. The valley test is defined in Eq. (19)~(20).

$$VT_1(pos_{check}) = \sum_{i=1}^{i=k_{vt}} ASD(pos_{check} - i) - k_{vt} \times ASD(pos_{check}) \quad (19)$$

$$VT_2(pos_{check}) = \sum_{i=1}^{i=k_{vt}} ASD(pos_{check} + i) - k_{vt} \times ASD(pos_{check}) \quad (20)$$

, where k_{vt} is the predefined range of the neighboring pixels for the valley test.

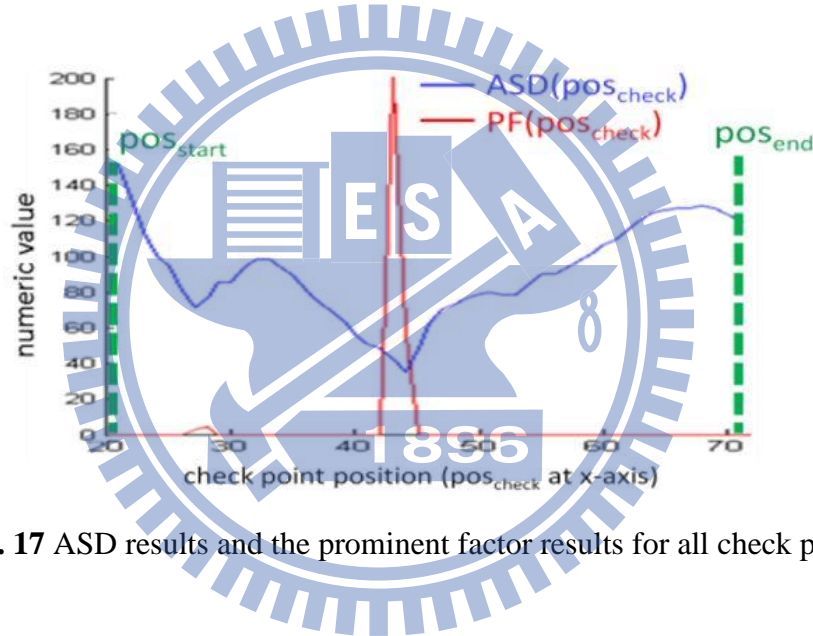


Fig. 17 ASD results and the prominent factor results for all check points

Only the check points passing the valley test (i.e., $VT_1(pos_{check}) > th_{vt}$ and $VT_2(pos_{check}) > th_{vt}$, where th_{vt} is a predefined threshold) are considered candidates for the symmetric center. In Fig. 17, it is obvious that ASD values decrease gradually toward the valley, and then increase gradually when leaving the valley. Therefore, the idea of calculating VT_1 and VT_2 is the same as calculating the triangular area in the range of k_{vt} , if the x-axis is shifted up to the ASD value of pos_{check} . The valley test threshold th_{vt} is designed by $\lfloor (k_{vt} \times k_{vt}) / 2 \rfloor$, which represents the area of an isosceles right triangle with side length k_{vt} . Empirically, the value of k_{vt} is 5 and the threshold th_{vt} will be 12. Based on the results of the experimental trials, the

threshold can provide satisfactory results. In Fig. 17, the check points at x-axis 27 and 43 pass through the valley test, so they are candidates of the symmetric center.

Due to local uniform intensities, the check point ($pos_{check} = 27$) is a valley, but it is not the actual symmetric center. To reject the false report, a prominent factor is defined as Eq. (21).

$$PF(pos_{check}) = 2 \times k_{vt} \times Int(pos_{check}) - \sum_{i=1}^{i=k_{pf}} Int(pos_{check} - i) - \sum_{i=1}^{i=k_{pf}} Int(pos_{check} + i) \quad (21)$$

, where k_{pf} is the predefined range for the prominent factor test.

A number of distinguishable objects, such as logos and license plates, commonly appear along the central line in the front region of cars. These objects differ obviously in intensity from the neighboring pixels and can be utilized to determine whether check points occur in the region of uniform intensity. The prominent factor sums the difference in intensity between $2 \times k_{pf}$ neighboring pixels and the check point. If the valley point has a sufficiently large prominent factor (i.e., $|PF(pos_{check})| > th_{pf}$, where th_{pf} is a predefined threshold), this check point is considered a candidate symmetric center of high priority. The prominent factor test works only for the check points that pass the valley test. If multiple candidates pass the prominent factor test, the candidate with minimum ASD is selected as the symmetric center of the current row. In the case in which no valley point passes the prominent factor test, the valley point with minimum ASD is selected as the symmetric center of the current row.

Because the tested car is individually identified with each specific target car model, the prominent factor threshold of a target car model can be defined differently. In an extreme case, a small threshold can be set if the target car model does not have distinguishable objects along the central line. In our experience, human eyes are capable of distinguishing two pixels if their difference in intensity exceeds 15. Therefore, in general cases for the front regions of cars, the prominent factor threshold th_{pf} is set $2 \times k_{pf} \times 15$. Empirically, the value of k_{pf} is 5; the threshold value is set to 150. Because the symmetric center and its neighboring pixels in the rear view

are not as obvious as those in the front view, the prominent factor threshold of the rear view is smaller than that of the front view.

In Fig. 17, the red curve shows the prominent factors of check points at 27 and 43. Because only the prominent factor of the check point at 43 is large enough, this check point is selected as the symmetric center of the current row. Next, a vote mechanism checks the occurrence frequency of the symmetric centers of all rows. The top 3 frequent symmetric centers are averaged to be the final result.

Through the above procedures, the reference position and the pose angle on the front (or rear) region of a tested car can be extracted. Since the side region is not horizontally symmetric, the symmetric center on the side region cannot be found. Because the side and front (or the side and rear) regions are segmented from the same car image, the pose angles of the side region and the front (or rear) region must be identical (i.e. $\theta_{test}^{side} = \theta_{test}^{front}$ (or θ_{test}^{rear}) = θ_{test}). Then, an obvious key point in an appearance part, e.g. a corner of a window, can be designated as the reference position. According to the set of template images for each target car model used, the distance between the designated reference position and the left end (in quadrant I or III) or the right end (in quadrant II or IV) with known pose angles (θ) can be measured in advance. The ratios of these distances to the width of the car also provide a monotonic function $Ratio_3(\theta)$ for each quadrant. Since the pose angle (θ_{test}) of the side region can be determined, the reference position of the side region can be predicted according to $Ratio_3$ and the position of the left or right end (i.e., $pos_{side} = pos_{left} + Ratio_3(\theta_{test})$ or $pos_{side} = pos_{right} - Ratio_3(\theta_{test})$). Therefore, if we can estimate the pose angle on the front (or rear) region accurately, the reference position and the pose angle on the side region can also be obtained accurately.

4.3 Experiments of car pose estimation

This dissertation evaluates the pose estimation accuracy of the approaches utilizing γ_{cov} and γ_{pose} through theoretical analysis and experimental results based on real data. The theoretical analysis below expands the two ratios by the Taylor series, and ignores the terms of the second and higher derivative. Equations (22)-(25) then evaluate the estimation errors of γ_{cov} , γ_{pose} , θ_{cov} , and θ_{pose} , denoted as $\Delta\gamma_{cov}$, $\Delta\gamma_{pose}$, $\Delta\theta_{cov}$, and $\Delta\theta_{pose}$, respectively.

$$\begin{aligned}\Delta\gamma_{cov} &= \frac{wid + \Delta wid}{hgt + \Delta hgt} - \frac{wid}{hgt} = \frac{wid}{hgt} \cdot \frac{(1 + \Delta wid/wid)}{(1 + \Delta hgt/hgt)} - \frac{wid}{hgt} \\ &\approx \frac{wid}{hgt} \cdot \left(1 + \frac{\Delta wid}{wid}\right) \cdot \left(1 - \frac{\Delta hgt}{hgt}\right) - \frac{wid}{hgt} \approx \frac{wid}{hgt} \left(\frac{\Delta wid}{wid} - \frac{\Delta hgt}{hgt}\right)\end{aligned}\quad (22)$$

$$\begin{aligned}\Delta\gamma_{pose} &= \frac{w_{half} + \Delta w_{half}}{hgt + \Delta hgt} - \frac{w_{half}}{hgt} = \frac{w_{half}}{hgt} \cdot \frac{(1 + \Delta w_{half}/w_{half})}{(1 + \Delta hgt/hgt)} - \frac{w_{half}}{hgt} \\ &\approx \frac{w_{half}}{hgt} \cdot \left(1 + \frac{\Delta w_{half}}{w_{half}}\right) \cdot \left(1 - \frac{\Delta hgt}{hgt}\right) - \frac{w_{half}}{hgt} \approx \frac{w_{half}}{hgt} \left(\frac{\Delta w_{half}}{w_{half}} - \frac{\Delta hgt}{hgt}\right)\end{aligned}\quad (23)$$

$$\Delta\theta_{cov} \approx \frac{\partial}{\partial \gamma_{cov}} (\text{Ratio}_{cov}^{-1}(\gamma_{cov})) \cdot (\Delta\gamma_{cov}) \approx \left. \frac{\Delta\theta_{cov}}{\Delta\gamma_{cov}} \right|_{\gamma_{cov}} \cdot \Delta\gamma_{cov}\quad (24)$$

$$\Delta\theta_{pose} \approx \frac{\partial}{\partial \gamma_{pose}} (\text{Ratio}_{pose}^{-1}(\gamma_{pose})) \cdot (\Delta\gamma_{pose}) \approx \left. \frac{\Delta\theta_{pose}}{\Delta\gamma_{pose}} \right|_{\gamma_{pose}} \cdot \Delta\gamma_{pose}\quad (25)$$

where Δwid , Δhgt , and Δwid_{half} are the estimation errors of wid , hgt , and wid_{half} . Normalize the heights of all tested cars to a fixed value, (i.e., $hgt = 112$). Since the estimation error of the bottom end is usually larger than that for the other ends, assume that $\Delta pos_{left} = \Delta pos_{right} = \Delta pos_{top} = 4$ pixels, and $\Delta pos_{bot} = 6$ pixels. Because the estimation error of the symmetric center is usually smaller than the errors of the four ends, assume that $\Delta pos_{sym} = 2$ pixels. Then, the estimation errors of wid , hgt , and wid_{half} will be 5.6, 7.2, and 4.4 pixels, respectively ($\Delta wid = \sqrt{4^2 + 4^2}$, $\Delta hgt = \sqrt{4^2 + 6^2}$, $\Delta wid_{half} = \sqrt{4^2 + 2^2}$).

Under the above assumptions, the error bars in Figs. 18(a) and 18(b) show the root mean square values of $\Delta\gamma_{cov}$ and $\Delta\gamma_{pose}$, respectively. Fig. 18(a) shows the root mean square (rms) values of $\Delta\theta_{cov}$ and $\Delta\theta_{pose}$. The pose error $\Delta\theta_{pose}$ is significantly smaller than $\Delta\theta_{cov}$ when the orientation of the tested cars exceeds 30° . Hence, the proposed pose estimation approach provides more accurate estimation results in the medium or large orientations, which are more difficult and important to deal with.

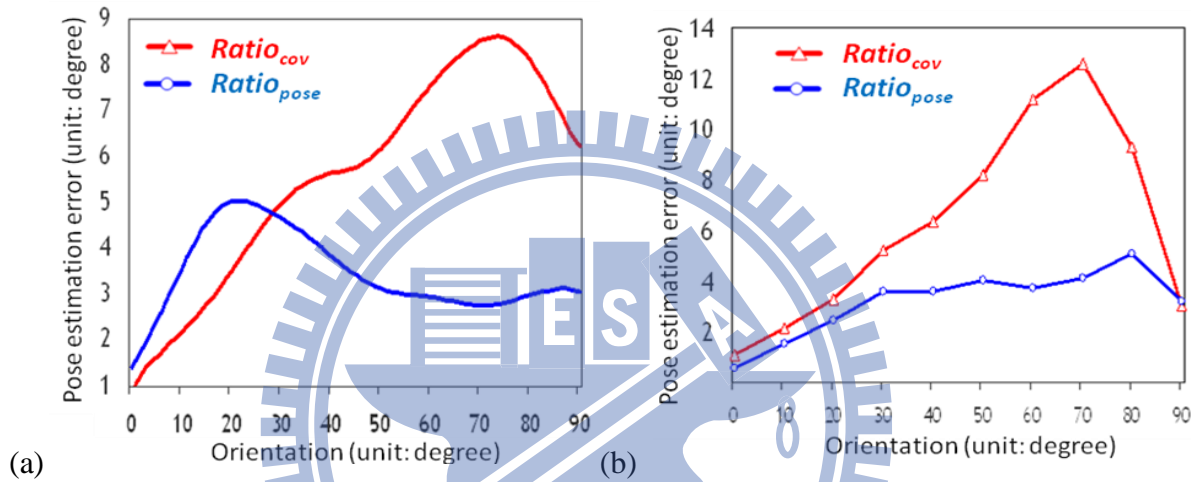


Fig. 18 The estimated pose errors of the conventional approach adopting γ_{cov} and the proposed approach adopting γ_{pose} on (a) theoretical analysis and (b) the experimental results

To compare the conventional and proposed approaches using real data, more than five hundreds car images captured in real traffic streams with various orientations $[0^\circ, 90^\circ]$ were tested. The car images are in the same size class and with similar shape. The ground truth of pose angle was first manually defined for each image. Fig. 18(b) shows the accuracies of the pose angle estimated by γ_1 and γ_2 . The pose errors in γ_1 range from 6.21° to 12.32° and γ_2 has smaller pose errors from 3.89° to 5.02° when the orientation is within $[40^\circ, 80^\circ]$. This phenomenon supports the theoretical analysis.

A comparison of Fig. 18(a) and Fig. 18 (b) shows that the real data performance is better than the theoretical analysis when the orientation is within $[0^\circ, 40^\circ]$. Because the more precise left end, right end, and the symmetric center are extracted within this orientation scope, the

proposed approach provides smaller and more stable pose error than the conventional approach for all orientations.

As the orientation of the tested car diverges more significantly, it becomes increasingly difficult to locate the symmetric center and to estimate the pose angle. However, the half width of the tested car varies more obviously in large pose angles than in small pose angles. In addition, along with the increment of the orientations, the check width becomes narrower and narrower, so the estimation error of the symmetric center is limited. These effects are capable of balancing the difficulty in extracting the symmetric center and estimating the pose angle in large orientation cases. Therefore, our proposed approach to pose estimation is able to estimate vehicle orientation accurately in most orientation cases.

However, when the orientation is very close to the lateral view (about $90^\circ \pm 10^\circ$ or $270^\circ \pm 10^\circ$), the symmetric center nearly disappears, thereby producing a higher degree of pose estimation error. In this extreme lateral view, the conventional pose estimation method using the ratio γ_{cov} is superior because the four ends can be extracted more accurately. A hybrid strategy is employed in the proposed system. When the estimated pose angle utilizing γ_{pose} falls within the range of $90^\circ \pm 15^\circ$ (or $270^\circ \pm 15^\circ$) and the pose angle utilizing γ_{cov} falls within the range of $90^\circ \pm 10^\circ$ (or $270^\circ \pm 10^\circ$), the estimated pose angle of our proposed approach will be replaced by the estimated pose angle of the conventional approach. This hybrid strategy makes the estimated pose angle more accurate in all orientations.

The proposed algorithm was implemented on an AMD Athlon, 2.0 GB RAM, 2.81 GHz PC in a Language C environment. The average computation time to find the symmetric center and estimate the rotation angle on a tested image (image size = 100×100 pixels) is less than 0.005 sec. This amount of computation time implies that the proposed initial pose estimation is applicable to most real-time systems.

Chapter 5 Car Size Classification Algorithm

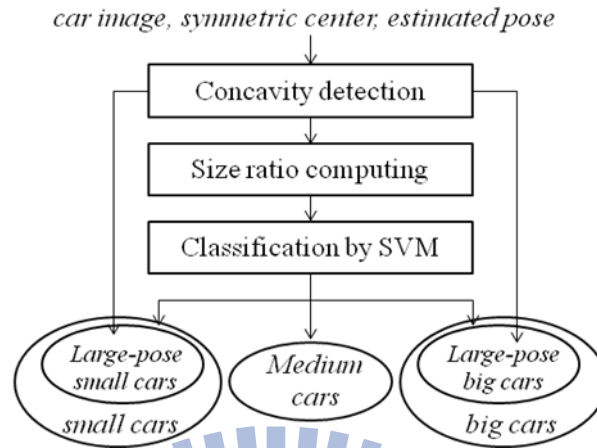


Fig. 19 Flowchart of the proposed size classification algorithm

After the ratio of pose estimation has been computed, another ratio to distinguish the size class of the car is also proposed in this chapter. Fig. 19 shows the main steps of the proposed size classification algorithm. When the car image with its estimated pose input, the concavity properties of big and small cars are utilized to separate them first. Chapter 5.1 shows the definition of the concavity properties. The size ratios of the remainder vehicles passing the check of concavity are further computed. The ratio is defined in Chapter 5.2. Then, the support vector machine is utilized to separate small, medium and big sizes of vehicles based on the size ratio in Chapter 5.3. The experimental results are provided in Chapter 5.4.

5.1 Concavity detection

Because the size classes of large pose vehicles are more difficult to be identified than the small-pose ones, most current feature-based size classification researches fix the capturing pose or limit the capturing poses into small orientations [27, 29]. Since the hoods of small-size cars is generally protruding, a concavity triangle full of background pixels can be found at the right top corner of the bounding box as shown in Fig. 20(a), if the orientation of

the vehicle falls in the quadrant II or IV. If the orientation falls in the quadrant I or III, the concavity triangle appears in the left top corner.

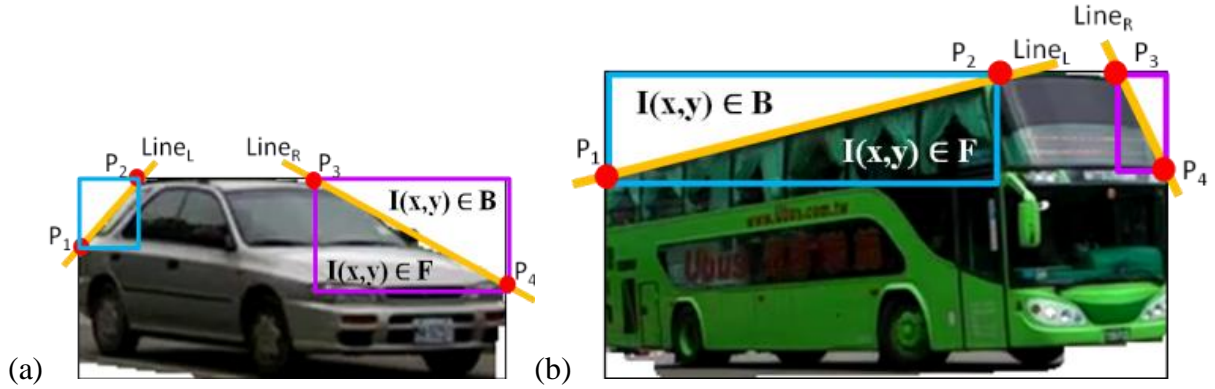


Fig. 20 The concavity triangles appeared in (a) a small car and (b) a big car with large poses

Due to the perspective effect, the big-size cars, such as tourist buses, reveal an up-slanted edge from fronts to rears [26] as shown in Fig. 20(b). The concavity triangle of a big car enclosed by the up-slanted edge appears in the opposite corner of a small car. According to the existence, the location, and the occupying area of the concavity triangles, the small and big-size vehicles with large poses can be discriminated.

For each vehicle image, we compute the intersection nodes: P_1 , P_2 , P_3 and P_4 , of the foreground pixels on the left, top and right boundary, as shown in Fig. 20. Compute the equations of the line segments: $line_L$ and $line_R$ passing through (P_1, P_2) and (P_3, P_4) , respectively, as Eq. (26). Eq. (27) and (28) define the existence scores of the left and right concavity triangles. If a background pixel (x, y) locates in the triangle above $line_L$ or $line_R$, the score increases. If a foreground pixel locates in the triangle below $line_L$ or $line_R$, the score also increases. Otherwise, the score decreases.

$$line_i : y = m_i x + b_i, \quad \text{where } m_i = \frac{p_2^y - p_1^y}{p_2^x - p_1^x}, \quad b_i = p_1^y - m_i p_1^x, \quad i \in \{L, R\} \quad (26)$$

$$score_i = \begin{cases} \sum_{x=pos_{left}}^{x \leq P_2^x} \sum_{y=pos_{top}}^{y \leq P_1^y} \frac{s_L}{(P_2^x - pos_{left}) \times (P_1^y - pos_{top})}, & i = L \\ \sum_{x=P_3^x}^{x \leq pos_{right}} \sum_{y=pos_{top}}^{y \leq P_4^y} \frac{s_L}{(pos_{left} - P_3^x) \times (P_4^y - pos_{top})}, & i = R \end{cases} \quad (27)$$

$$s_i = \begin{cases} 1, & \text{if } y < m_i x + b_i \text{ \& } I(x, y) \in B \\ 1, & \text{if } y \geq m_i x + b_i \text{ \& } I(x, y) \in F \\ -1, & \text{if } y < m_i x + b_i \text{ \& } I(x, y) \in F \\ -1, & \text{if } y \geq m_i x + b_i \text{ \& } I(x, y) \in B \end{cases}, \quad i \in \{L, R\} \quad (28)$$

When the vehicle orients anticlockwise, if the existence score of the right concavity triangle is larger than a predefined threshold ($th_1 = 0.9$, empirically), and the width and height of the triangle are both larger than a predefined threshold ($th_2 = 0.3 \times h$, empirically), the vehicle is classified into small size. Otherwise, the existence score of the left concavity triangle is checked. If the existence score, the width, and the height of the triangle are larger than the respective thresholds, the vehicle is classified into big size.

5.2 Size ratio computing

Conventionally, the aspect ratio of the bounding box as Eq. (14) can be utilized to estimate the vehicle pose or classify the vehicle size. However, the ratio can accurately classify the vehicle size only when the pose is fixed, or estimate the vehicle pose correctly when the vehicle is limited to the same size class. A detected symmetric center can be utilized to estimate the head width ($wid_{head} = 2 \times wid_{half}$) of the car. Similar to the definition of the pose ratio as Eq. (16), a size ratio of the boundary height to head width is defined as Eq. (29).

$$\gamma_{size} = \frac{hgt}{2 \times wid_{half}} = \begin{cases} \frac{pos_{bot} - pos_{top}}{2 \times (pos_{sym} - pos_{left})}, & \text{if } \theta \in [0^\circ, 90^\circ] \text{ or } [180^\circ, 270^\circ] \\ \frac{pos_{bot} - pos_{top}}{2 \times (pos_{right} - pos_{sym})}, & \text{if } \theta \in [90^\circ, 180^\circ] \text{ or } [270^\circ, 360^\circ] \end{cases} \quad (29)$$

Fig. 21 shows the distribution of the ratios (r_{size} , r_{pose}) of small, medium and big size cars. 1000 small-size, 1000 medium-size and 1000 big-size car images are tested and the orientations of the tested cars are of variety. Since very little intersections exist between the points of small, medium and big cars, very few cars are misclassified and higher size classification accuracy can be anticipated.

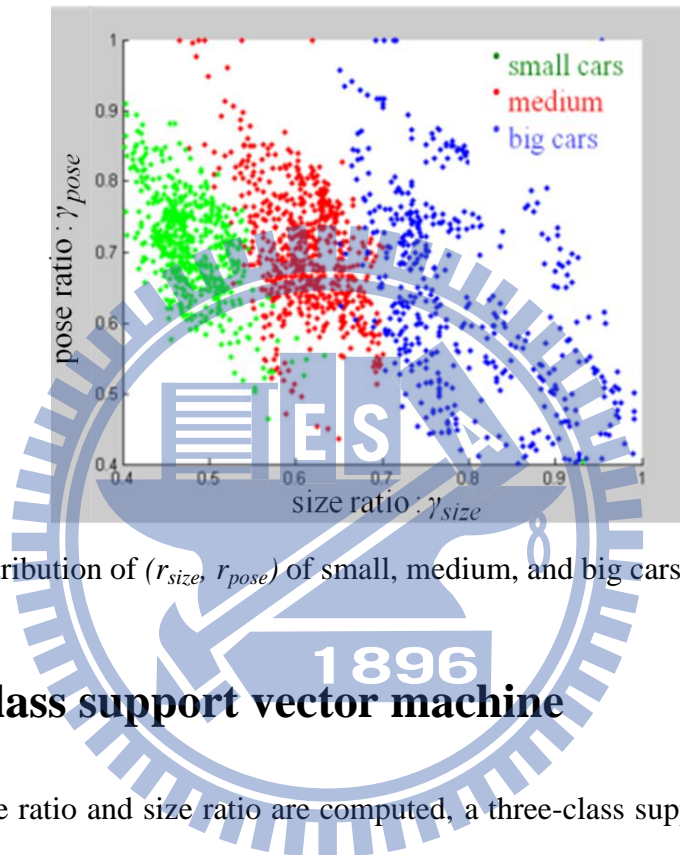


Fig. 21 Point distribution of (r_{size} , r_{pose}) of small, medium, and big cars across various poses

5.3 Three-class support vector machine

After the pose ratio and size ratio are computed, a three-class support vector machine is designed to classify the computed aspect ratios into small, medium and big classes. Support vector machine is an effective classification and pattern recognition tool. Given a set of feature vectors which belong to multiple classes, a SVM finds the hyper-plane maximizing the distance of classes [53]. Because SVM provides excellent capability for identifying features, hence, it is popular for classifying vehicle colors [7], poses [55] and models [29].

One hundred vehicle images are collected for each size class to train the support vector machine. For each image, the boundary ends and symmetric center are manually labeled and the pose and size ratios are computed as the features used in the support vector machine. Since the number of features is very small, the computation load is slight.

5.4 Experiments of car size classification

To test the proposed system on vehicle size classification, this study collects a vehicle image dataset. The dataset captures car images from real traffic streams obtained by cameras. The database contains 1000 small-size cars, 1000 medium-size cars and 1000 big-size cars. Vehicles with various colors and models are included in the dataset. The dataset does not include vehicles which do not satisfy the symmetric property, such as SUV and jeeps. In addition, overloaded pickups are also not included in the dataset, since that the ratios may be incorrectly calculated.

To understand the classification accuracy under different orientation ranges, the images are manually divided into three subsets: $view_1$, $view_2$ and $view_3$. In $view_1$, the vehicles approach to frontal view and their rotation angles are in the range of $[0^\circ, 10^\circ)$. In $view_2$, the vehicles have rotation angles in the range of $[10^\circ, 25^\circ)$. Vehicles with rotation angles larger than 25° are collected in the $view_3$ subset.

Table 7 Accuracy of the concavity detection

	<i>(a) Large pose small car detection</i>			<i>(b) Large pose big car detection</i>		
	<i>small(%)</i>	<i>medium(%)</i>	<i>large(%)</i>	<i>small(%)</i>	<i>medium(%)</i>	<i>large(%)</i>
$view_1$	5.8	0	0	0	0	0
$view_2$	38.24	0.42	0	0.08	0	10.43
$view_3$	94.55	0.63	0	0.54	1.27	78.62

The small and big cars with large poses are detected by the concavity property. Table 7 shows the detection accuracy in different views. In Table 7(a), most (94.55%) small cars in $view_3$ and 38.24% of vehicles in $view_2$ are correctly identified to small size cars. Only 0.42% of medium cars in $view_2$, and 0.63% of medium cars in $view_3$ are misclassified to small cars by the concavity property. Table 7(b) shows the percentage of vehicles classified to big cars

by the concavity property. Over 78% of big cars in $view_3$ and 10% of big cars in $view_2$ are correctly identified. Very few small and medium cars are misclassified to big cars.

The accuracies of the detected symmetric centers in different sizes and different views are shown in Table 8. For each image, the height is normalized to 100 pixels, and we manually label the symmetric center on the image as the ground truth. If the distance between the detected center and the ground truth is larger than 10 pixels, it is considered as a false detection. Table 8(a) shows the percentage of the false detection images and Table 8(b) shows the average distance (bias) of the detected symmetric centers to the ground truth.

In Table 8(a), there are only 2.03% of vehicles whose symmetric centers are far from the ground truth. The false detection of $view_3$ is a little higher than the other views. In Table 8(b), since vehicles with large poses are excluded in advance, the symmetric center detection algorithm performs well on the remainder vehicles. The average bias of the detected symmetric center is less than 2.16 pixels. This small bias generates accurate enough aspect ratio to classify size classes.

Table 8 Accuracy of the symmetric center detection

	<i>(a) Percentage of false detection</i>			<i>(b) Bias of symmetric center</i>		
	<i>small(%)</i>	<i>medium(%)</i>	<i>large(%)</i>	<i>small(px)</i>	<i>medium(px)</i>	<i>large(px)</i>
$view_1$	0	0	1.06	1.32	0.92	1.15
$view_2$	0	1.78	0.81	1.94	1.68	1.53
$view_3$	6.67	1.91	5.05	4.38	1.88	4.56

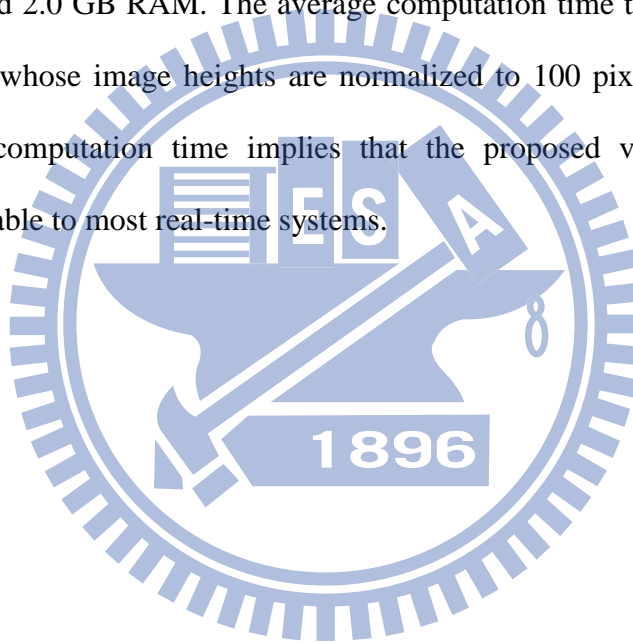
After the symmetric center is located, the aspect ratios: γ_{cov} , γ_{size} , and γ_{pose} are computed, and the support vector machine is used to classify the vehicle size by the ratios. Conventional approaches [19, 21] utilize the aspect ratio: γ_{cov} of bounding box to classify the vehicle size. Due to the various orientations, the ratio can only achieve classification accuracy to 73.6%, 69.2% and 64.8% for small, medium and big cars, respectively, as shown in Table 9. By adopting the ratios of (γ_{size} , γ_{pose}), the classification accuracies are significantly improved

to 95.6%, 90.9% and 87.7%, for small, medium and big cars, individually.

Table 9 Accuracy of the size classification

<i>T/D</i>	<i>small</i>		<i>medium</i>		<i>big</i>	
	$svm(r_{cov})$	$svm(r_{size}, r_{\theta})$	$svm(r_{cov})$	$svm(r_{size}, r_{\theta})$	$svm(r_{cov})$	$svm(r_{size}, r_{\theta})$
<i>small</i>	736	956	131	61	37	6
<i>medium</i>	264	37	692	909	315	117
<i>big</i>	0	7	177	30	648	877
<i>accuracy</i>	73.6%	95.6%	69.2%	90.9%	64.8%	87.7%

The proposed algorithm is implemented in C and deployed on a PC with AMD Athlon 2.81 GHz CPU and 2.0 GB RAM. The average computation time to classify the vehicle size on tested images, whose image heights are normalized to 100 pixels, is less than 0.01 sec. This amount of computation time implies that the proposed vehicle size classification approach is applicable to most real-time systems.



Chapter 6 Car Model Recognition Algorithm

Although SVM is able to classify vehicle size, it can only distinguish vehicles with distinct shape difference. To identify cars with similar shape, template matching technique should be adopted and a mirror morphing scheme [56] is proposed to accommodate the severe pose variation problem. Mirror morphing requires knowing the locations of shape feature of cars first. Chapter 6.1 introduces the processing of the shape feature extraction. Mirror morphing is modified from the general morphing technique [57]. General morphing and mirror morphing are introduced in Chapters 6.2 and 6.3, respectively. Finally, the synthesized image of the car is matched against the template image. The procedure of template matching is drawn in Chapter 6.4. The experimental results are provided in Chapter 6.5.

6.1 Shape feature extraction

When the view of the tested car is angled, the appearances vary significantly in size, shape, and geometric relationship. To correctly determine the locations of the shape features at any orientation becomes critical and challenging. The shape feature extraction algorithm must show flexibility and robustness in its ability adaptive to the geometric deformation. It must also exhibit controllability to prevent from searching in the erroneous directions. ASM (Active Shape Model) [36][58] is used to locate key points of the shape feature on cars.

ASM comprises two phases: the off-line training phase and the on-line searching phase. In the off-line phase, several training image sets are prepared for each target model. Each training image set contains 4 or 5 images and covers a specific orientation range. On each training image, a number of key points, such as corners, or boundaries of shape features are marked manually. The coordinates of these key points are collected as a shape vector. Each training set generates a set of shape vectors and computes a mean shape vector. The principal

eigenvectors of these shape vectors are calculated to form a geometry variation matrix. The matrix controls the deformation of the mean shape vector by providing the weight for each principal eigenvector. The weight values are varied according to the shape deformation and the deformation is constrained within a suitable limit.

In the on-line phase, ASM deforms the mean shape vector of the selected training image set to search for the corresponding key points on the tested car image iteratively as follows:

$$sv_n = sv_n + MV \times wv_{ASM} \quad (30)$$

, where sv_n is the estimated shape vector of the tested car in the n^{th} iteration. MV is the geometry variation matrix. wv_{ASM} is the weighting vector.

The initial shape vector sv_0 can be represented by two parts: the center position and the orientation of the initial shape vector. The two parts can be acquired by the symmetric center (pos_{cen}) and the estimated pose angle (θ_{pose}) extracted by $Ratio_{pose}$. After the training set which covers θ_{pose} is selected, the geometry variation matrix computed from the selected training set is determined. Each key point on sv_0 moves toward the closest edge on the tested image. The movement is constrained by the geometry variation matrix and the weighting vector. Thus, the shape extracted by ASM on the tested car image is constrained to a shape similar to that of the template. When the ASM searching process converges, the corresponding key points on the tested car image can be located and the shape features of the tested car have been extracted.

Basically, the search process of ASM is sensitive to the initial shape. If the center position is too distant from the correct corresponding position, ASM may search in the wrong direction, so that most key points are incorrectly located. In Fig. 22(a), the yellow lines which indicate contours connecting the key points (painted in red) located by ASM, deviate from the actual shape of the tested car. Even though the center position of the initial shape vector is correct, ASM would not guarantee to determine the correct shape of the tested car. In Fig. 22(b), if an improper training image set is selected, the orientation of the initial shape vector

might quite deviate from the orientation of the actual shape vector. The geometry variation matrix which controls the searching direction may also be erroneous. Thus, to locate the key points of the tested car by ASM more correctly as in Fig. 22(c), it is crucial to properly establish the center position of the initial shape vector and the training image set for the ASM searching process.

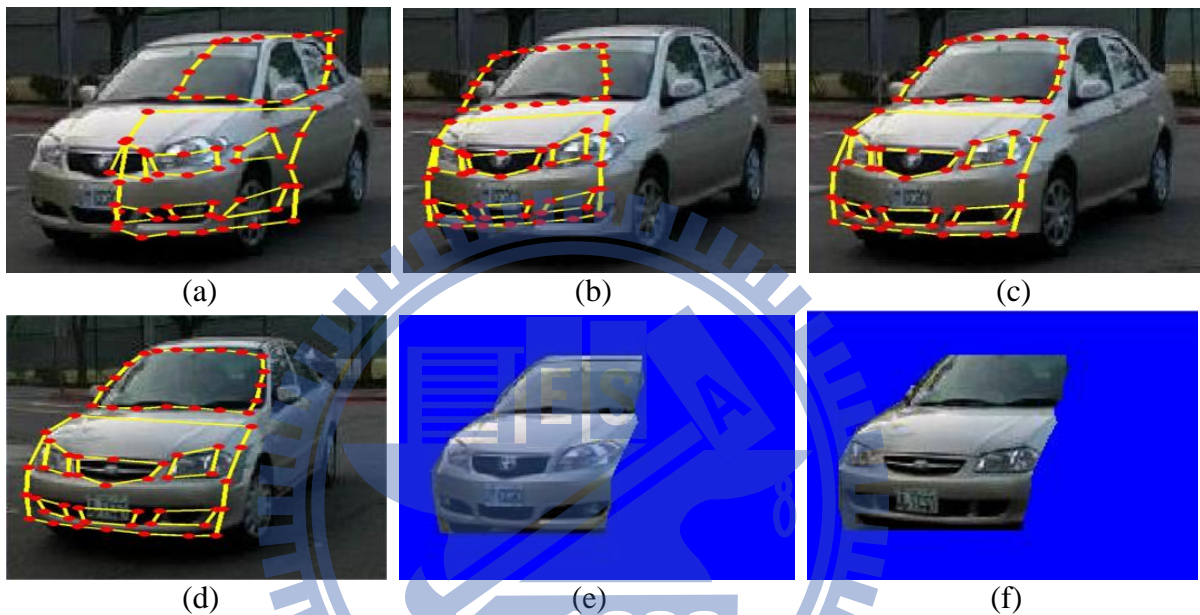


Fig. 22 ASM result (a) with erroneous initial position, (b) with improper initial mean shape, (c) with proper initialization setting. (d) ASM result of a different model car with the template, (e) the segmented result of (c), and (f) the segmented result of (d)

Fig. 22 (c)~(d) show two cases in which we operate ASM for car model recognition. Consider a Toyota VIOS as the target car. In Fig. 22 (c), the key points on a tested car image of the Toyota VIOS are correctly searched by ASM. In Fig. 22 (d), a tested car image that is not Toyota VIOS is processed by ASM. ASM also accurately located the corresponding key points on this tested car image without any indication that the two vehicles are not the same model. This implies that utilizing ASM alone has difficulty distinguishing one particular model of cars from other similar models. Although ASM is a good tool for extracting shape feature from car images, it does not usually perform well on car model recognition. The

mirror morphing process described later could help to accomplish the mission. From the ASM results, we separated (segmented) the front regions of cars in Fig. 22 (c)~(d) and their segmented images are shown in Fig. 22 (e)~(f), respectively.

6.2 General morphing

The general morphing technique [57] is the process of transforming two source images (Img_1, Img_2) into a target image (Img). Each pixel in the source images will be transformed (prewarped) to the same coordinate basis. Then, the target image is generated by a weighted combination of the source images controlled by a parameter λ as follows.

$$(x_2, y_2) = COR(x_1, y_1) \quad (31)$$

$$(x, y) = \lambda \times (x_1, y_1) + (1 - \lambda) \times (x_2, y_2) \quad (32)$$

$$Img(x, y) = \lambda \times Img(x_1, y_1) + (1 - \lambda) \times Img(x_2, y_2) \quad (33)$$

, where COR defines the corresponding relationship between a pixel (x_1, y_1) in Img_1 and a pixel (x_2, y_2) in Img_2 . The function $Img(x, y)$ is the intensity at the pixel (x, y) in the target image Img .

The prewarping process requires the correspondence relationship between Img_1 and Img_2 for all pixels. If we only acquire the correspondence relationship for some key points, not all pixels, the Beier-Neely's Field algorithm [59] can be applied. Fig. 23 shows an example to briefly illustrate the Beier-Neely's Field algorithm.

Fig. 23 shows the key points on the source images: Img_1 and Img_2 . The function (f) represents the correspondence between these key points. For example, the key point P_6 in Img_1 corresponds to the key point P'_{10} in Img_2 . For a pixel S as shown in Fig. 23 (c), which is in Img_1 but not included in the known key points, we acquire its corresponding pixel S' as shown in Fig. 23 (d) on Img_2 . The nearest point R to S along line segment $P_m P_n$ ($m=6, n=7$ in

Fig. 23 (a)) is obtained. The distance from S to R is denoted by v and the distance from P_m to P_n is denoted by u . These distances can be computed by Eq. (34). Function $Perp$ returns the vector perpendicular to the input vector. Because S' corresponds to S , S' will be located at v pixels away from the line segment $P'_{f(m)}P'_{f(n)}$ ($f(m)=10, f(n)=9$ in Fig. 23 (b)) as S away from the line segment P_mP_n . Hence, S' in Img_2 can be computed by Eq. (35).

$$u = \frac{(S - P_m) \cdot (P_n - P_m)}{\|P_n - P_m\|^2}, \quad v = \frac{(S - P_m) \cdot Perp(P_n - P_m)}{\|P_n - P_m\|^2} \quad (34)$$

$$S' = P'_{f(m)} + u \cdot (P'_{f(n)} - P'_{f(m)}) + \frac{v \cdot Perp(P'_{f(n)} - P'_{f(m)})}{\|P'_{f(n)} - P'_{f(m)}\|^2} \quad (35)$$

When multiple pairs of line segments are considered, the weight of each corresponding line segment needs to be calculated. The weight is determined by the distance from S to each line segment P_mP_n . The closer the pixel to the line segment, the higher the weight is.

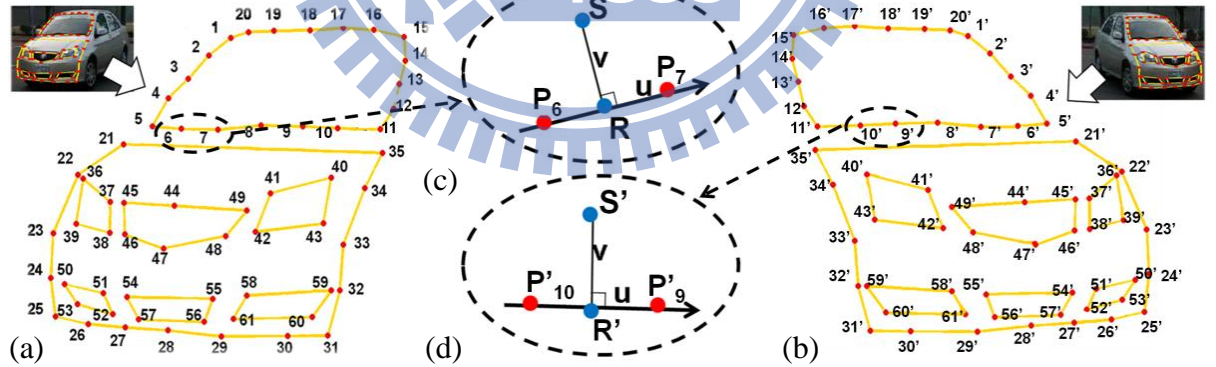


Fig. 23 Establish the corresponding relationship between Img_1 and Img_2 for all pixels based on known key point pairs (a) source image Img_1 with 61 key points (b) source image Img_2 with 61 key points (c) a pixel S in Img_1 but not one of 61 key points (d) the pixel S' in Img_2 corresponding to the pixel S in Img_1

6.3 Mirror morphing

Next, we utilize a mirror morphing technique to synthesize a segmented car image on the front (or rear) region into a front (or rear) view image. Since the horizontal symmetry does not exist in the side region, a quasi mirror morphing is designed. The quasi mirror morphing is able to transform the segmented car image on the side region into an approximate typical side view image. The mirror morphing and the quasi mirror morphing are introduced as follows in order.

Because a car is generally symmetric horizontally in the front (or rear) region, for any tested car image Img_1 segmented from the front (or rear) region, an opposite view image Img_2 can be produced by mirror transformation as follows.

$$(x_2, y_2)_i = (Width(Img_1) - x_1, y_1)_i \quad (36)$$

$$Img(x_2, y_2)_i = Img_1(Width(Img_1) - x_1, y_1)_i \quad (37)$$

, where $Width(Img_1)$ represents the width of Img_1 . The opposite view image termed the mirror image is adopted as the other source image Img_2 for morphing.

Since Img_2 is produced by Img_1 , the correspondence relationship between the key points in Img_1 and Img_2 can be acquired if the shape vector is defined to be symmetric. For example, if Img_2 in Fig. 23 (b) is the mirror image of Img_1 in Fig. 23 (a), the key points ($P_{36} \sim P_{39}$) on the left car lamp in Fig. 23 (a) are transformed into the key points ($P'_{36} \sim P'_{39}$) on the right car lamp in Fig. 23 (b). The transformed right car lamp ($P'_{36} \sim P'_{39}$) in Img_2 should be morphed with the right car lamp ($P_{40} \sim P_{43}$) in Img_1 . Therefore, if the correspondence relationship between the key points ($P_{36} \sim P_{39}$) and ($P_{40} \sim P_{43}$) in Img_1 has been known, the correspondence relationship between the key points ($P_{40} \sim P_{43}$) in Img_1 and ($P'_{36} \sim P'_{39}$) in Img_2 is acquired.

We define the shape vector (i.e., landmark the locations of key points) of ASM to be symmetric. Each key point $(x_1, y_1)_i$ in Img_1 has its corresponding key point $(x_1, y_1)_{Cor(i)}$ in

the opposite side in Img_1 . For example, P_1 corresponds to P_{15} and P_5 corresponds to P_{11} . (i,j) denotes the index of key points as a corresponding pair, a function Cor is denoted as Eq. (38).

$$j = Cor(i) \quad (38)$$

According to Eq. (36)~(37), all key points in the mirror image Img_2 is generated by the key points in the image Img_1 . By knowing the corresponding pairs of the key points in Img_1 , the correspondence between the key point $(x_2, y_2)_i$ in Img_2 and the key point $(x_1, y_1)_{Cor(i)}$ in Img_1 can be acquired. For example, P'_1 in Fig. 23 (b) is generated by P_1 (i.e., $P'_1 = Width(Img_1) - P_1$) and P_1 corresponds to P_{15} in Fig. 23 (a), so P'_1 corresponds to P_{15} . After the correspondences between all the key points in Img_1 and Img_2 are known, the Beier-Neely's Field algorithm [59] can be utilized to extend the correspondences from these key points to all pixels. Finally, the synthesized image can be produced as follows:

$$\begin{aligned} (x, y)_{Cor(i)} &= \lambda \times (x_1, y_1)_{Cor(i)} + (1 - \lambda) \times (x_2, y_2)_i \\ &= \lambda \times (x_1, y_1)_{Cor(i)} + (1 - \lambda) \times (Width(Img_1) - x_1, y_1)_i \end{aligned} \quad (39)$$

$$\begin{aligned} Img(x, y)_{Cor(i)} &= \lambda \times Img_1(x_1, y_1)_{Cor(i)} + (1 - \lambda) \times Img_2(x_2, y_2)_i \\ &= \lambda \times Img_1(x_1, y_1)_{Cor(i)} + (1 - \lambda) \times Img_2(Width(Img_1) - x_1, y_1)_i \end{aligned} \quad (40)$$

Fig. 24(a) is the same image of Fig. 22(e) and Fig. 24(b) is the mirror image of Fig. 24(a). The correspondence relationship between Fig. 24(a) and 24(b) is constructed by their ASM results and the images are synthesized by general morphing [57] into Fig. 24(c). Since mirror morphing is able to transform the tested car image into its typical front (or rear) view, the pose variation and center bias is eliminated. Although the error of locating shape features of the tested car may make the synthesized image a little blurred, the blurred phenomenon will not degrade the recognition performance as much as the effects of pose error and center bias.

To synthesize the segmented image of the side region into an exact lateral view (90° or 270°), a mirror image orientated to the opposite lateral view is required. However, this mirror

image cannot be generated using mirror transformation because horizontal symmetry does not exist in the side region. In this study, if the estimated pose angle of the tested car is θ^{est} , a template image with pose angle $180^\circ - \theta^{est}$ (or $540^\circ - \theta^{est}$) is selected as the mirror image. For example, because the estimated pose angle of the tested car in Fig. 24(e) is 70° , the template with the pose angle of 110° shown in Fig. 24(f) is selected to be the mirror image.



Fig. 24 (a) Segmented image of front region, (b) mirror image of (a), (c) synthesized car image from (a) and (b) by mirror morphing, (d) synthesized template car image, (e) segmented image of side region, (f) selected mirror template, (g) synthesized car image from (e) and (f) by quasi mirror morphing

The shape features of the segmented image on the side region and its mirror image can be extracted by ASM in the shape feature extraction step. The shape features referring to the same characteristics are in correspondence. Then, general morphing is performed on the segmented side region image (e.g. Fig. 24(e)) and the selected template image (e.g. Fig. 24(f)). An approximate lateral view image is synthesized as shown in Fig. 24(e). This synthesizing procedure for the side region image is termed as the quasi mirror morphing process.

Since the mirror image is not produced from the segmented side region of the tested car, the quasi mirror morphing process can reduce but not eliminate the pose error as the mirror morphing does in the front or rear region. Fortunately, shape deformations in situations close to the lateral view are generally less serious than those found in other views. Assume $\Delta\theta$ is the variation of pose angle of a tested car. In the lateral view, deformations are mainly due to the

width of the car multiplying $\sin(\Delta\theta)$ and the length of the car multiplying $1-\cos(\Delta\theta)$. While, in the front or rear view, the deformations are mainly due to the width multiplying $1-\cos(\Delta\theta)$ and the length multiplying $\sin(\Delta\theta)$. Since $\sin(\Delta\theta)$ is obviously larger than $1-\cos(\Delta\theta)$ when $\Delta\theta$ is small and the car length is much larger than the car width, the shape deformations of cars in the lateral view would be smaller than those in the front or rear view. This effect could increase the tolerance on pose error in the neighborhood of the lateral view when we recognize car models by template matching. Hence, the quasi mirror morphing process can still provide good recognition performance in cases close to the lateral view.

6.4 Template matching

The morphed tested car image (Fig. 24(c)) is matched against the morphed template car image (Fig. 24(d)) to recognize the car model. For each target model, the morphed template with a pose angle closest to the morphed tested car is selected from the template database. The SAD (Sum of absolute differences) method calculates the sum of absolute difference of r, g, and b channels for each pixel between the tested car and the template car, and averages over all pixels on the tested car. The SAD-score is referred to the matching error of the tested car image. If the matching error is smaller than a predefined threshold th_{model} , the tested car and the template car are considered to be of the same model; otherwise, they are considered to be different model cars. The threshold is defined by the number of non-target model cars that tolerant to misclassification as the target model. The more the misclassification of non-target model cars can be tolerated, the larger the threshold is.

We consider the scenario that the cars with the same shape but different colors to be different models, so that the tested car which has a different color to that of the template car will have large SAD-score and can be easily identified to be different models. The SAD-score is calculated after intensity normalization. The intensity normalization contains computing the average intensity of the template car first and then adjusting the average intensity of the tested

car to the average intensity of the template car. After intensity normalization, the SAD approach is able to accommodate general lighting and weather variations.

We compared the SAD technique with a number of matching methods, such as NCC (normalized cross correlation) and the edge-based techniques [60]. The NCC method can provide slightly superior recognition accuracy than the SAD method, but, the computation load of NCC is much heavier than that of SAD. The edge-based method can eliminate the influence of vehicle colors, while the improvement from the edge-based method in the scenario that we are investigating is quite limited. Furthermore, the edge-based method usually loses some texture information and generates significant errors from falsely detected edges. Therefore, the edge-based method generally obtains worse car recognition rate than the SAD method. Since the SAD method shows robust performance and efficient computation, the SAD method is adopted in the template matching step of the proposed system.

Fig. 25 illustrates the matching errors by algorithms with and without mirror morphing (with_MM/ no_MM) with respect to various pose estimation errors (from 0° to 90°) and center biases (0 and 5 pixels) in the initial phase. The tested cars that are the same models as the templates are referred to as target model cars (Tar). The tested cars that are different models to the templates are called non-target model cars (NTar). The algorithm is tested on target model cars, and the matching errors shown in Fig. 25 are the statistical results over numerous tested car images. Each tested car image has a SAD-score and the RMS (Root Mean Square) value of all the computed SAD-scores is calculated. Since non-target model cars are utilized to determine the threshold for recognition, the matching error associated with the non-target model cars in Fig. 25 shows the minimum SAD over all non-target car images. This minimum value can be considered the threshold. The target model cars with their SAD-scores lower than the threshold are considered correct recognitions. Therefore, as shown in Fig. 25, if the (RMS) matching error of an algorithm for all target model cars is obviously

smaller than the minimum SAD for the non-target model cars, the performance of the algorithm is considered good.

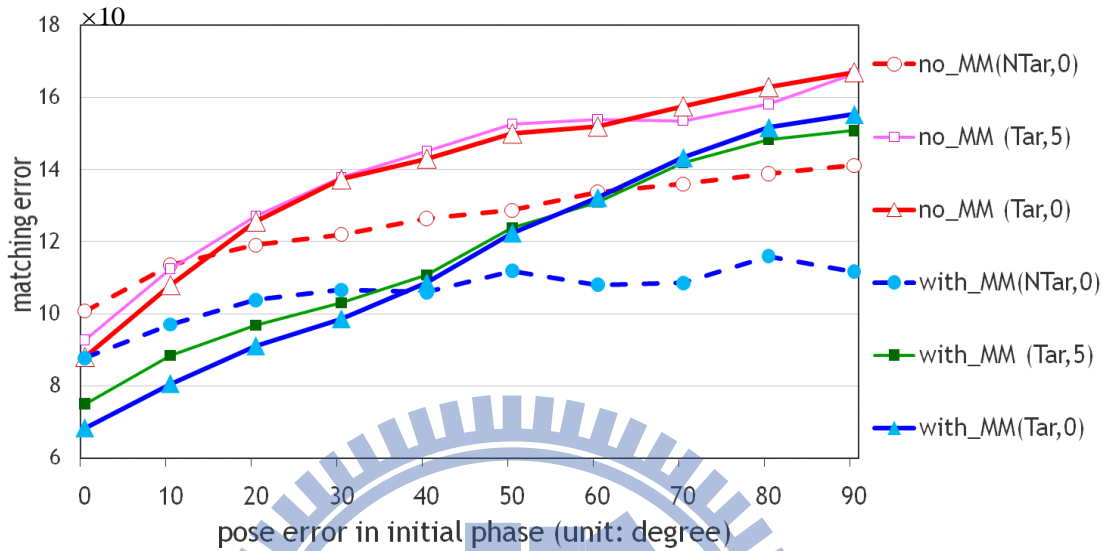


Fig. 25 Sensitivities of the algorithms with and without mirror morphing to pose error and center bias in the initial phase

As seen in Fig. 25, even in ideal conditions with no center bias, the algorithm without mirror morphing (no_MM(Tar,0)) provided only moderate performance when the pose estimation error is less than 10°. In conditions in which the center bias is 5 pixels, the algorithm without mirror morphing (no_MM(Tar,5)) achieves only moderate performance when the pose estimation error is smaller than 3°. Such requirements are extremely difficult to achieve. However, when the center bias is zero, the algorithm with mirror morphing (with_MM(Tar,0)) can work well when the pose estimation error is smaller than 30°, and in cases in which the center bias is 5 pixels, the algorithm with mirror morphing (with_MM(Tar,5)) still works well even when the pose estimation error is less than 20°. Such requirements can be easily achieved in most practical situations. It can be concluded that mirror morphing is an effective technique for the recognition of car models and other applications dealing with severe variations in pose.

6.5 Experiments of car model recognition

6.5.1 Experimental description

Three algorithms: PETM, PASM, and ASMM are presented to demonstrate that the proposed mirror morphing provides a considerably improvement to the traditional pose estimation and matching methods. Subsequently, a different type of algorithm, QRAFM [35] is implemented and compared with the proposed algorithm. The algorithms are described as follows:

1) PETM (Pose Estimation and Template Matching)

PETM performs the pose and center estimation first and then selects a template car image which covers the estimated angle. All the tested car images are offset to the center position of the selected template and then matched against the selected template. PETM can represent the category of a baseline car model recognition method [34] utilizing pose estimation and template matching.

2) PASM (Pose Estimation and ASM)

PASM performs ASM to extract shape features after the pose angle have been estimated. The center position will be re-estimated by the center of the shape features, and a better alignment for template matching is acquired. The segmentation processing is also performed and the tested car image is matched against the template which covers the estimated pose angle. PASM can represent the category of an improved car model recognition method [39] based on pose estimation and template matching.

3) ASMM (ASM and Mirror Morphing)

ASMM is our proposed algorithm which performs the whole set of functions of five steps. Comparing with the algorithm PASM, mirror morphing is performed for both tested and template car images. Then, the morphed tested car image is matched against the morphed template car image which covers the estimated pose angle. In PASM and ASMM, the

front (or rear) view and the side view images of the tested car are extracted separately and they are matched against their individual templates.

4) QRAFM (Quasi-Rigid Alignment and Flexible Matching)

The algorithm [35] adopts quasi-rigid alignment instead of a pose estimation technique. The tested car image is aligned to the template image based on the established corresponding relationships between the key segments on the tested cars and the template. Subsequently, a local flexible matching mechanism is designed to divide the tested car image into multiple patches, and each patch is shifted in the neighboring regions to obtain the optimal match to the template.

Since the procedures of PETM, PASM and ASMM are similar, more detailed analyses are provided to demonstrate the contribution when the mirror morphing technique is involved. Because the accuracies of the estimated pose angle and center position considerably affects the performance comparison, two estimation approaches, introduced in Chapter 4, are tested. The conventional pose estimation approach based on the ratio of car width to car height is denoted as A, and our proposed pose estimation approach based on the ratio of half front (or rear) width to car width is denoted as B. The PETM, PASM, and ASMM algorithms can use either A or B as the initial setting to analyze the intermediate data. The names of these algorithms and their respective functionality are illustrated in Table 10. Because the QRAFM algorithm adopts different strategies for recognition, the initial setting is not applied and only the recognition performance is compared.

A car image database is constructed containing over six thousand car images with orientations from 0° to 360° . The images were captured in real traffic streams and some examples are shown in Table 11. The images in the database are categorized as target model and non-target model. Two car models: Toyota VIOS in silver and Nissan New SENTRA in



yellow, are selected as the target car models. All the non-target model cars are similar in shape and color to the target model cars.

Table 10 The functions working in the car model recognition algorithms

Step Algo. Name	(1) Pose & Center Estimation	(2) Shape Feature Extraction	(3) Car Region Segmentation	(4) Mirror Morphing	(5) Template Matching
PETM-A	△	×	×	×	○
PETM-B	◆	×	×	×	○
PASM-A	△	○	○	×	○
PASM-B	◆	○	○	×	○
ASMM-A	△	○	○	○	○
ASMM-B	◆	○	○	○	○
△: the conventional pose estimation approach ◆: our proposed pose estimation approach. ×: the step is not performed ○: the step is performed					

In the dataset, 36 car images at $0^\circ, 10^\circ, 20^\circ, 30^\circ, \dots$ and 350° for each target model are collected as the template images. In the pose and center estimation step, the template images are divided into four sets, $[0^\circ, 90^\circ], [90^\circ, 180^\circ], [180^\circ, 270^\circ]$, and $[270^\circ, 360^\circ]$. For each set, ten template images are utilized to construct a monotonic function to estimate the pose of a tested car. In the shape feature extraction step, ASM requires 36 training sets to train the active shape models for all sample pose angles ($\theta = 0^\circ, 10^\circ, \dots, 350^\circ$). Each training set collects the required template images as a set, covering the following pose angles: $\theta-20^\circ, \theta-10^\circ, \theta, \theta+10^\circ$, and $\theta+20^\circ$ for the sample pose angle θ . In the mirror morphing step, the side regions of all 36 template images are segmented for quasi-mirror morphing. In the template matching step, all 36 template images are morphed and one of the morphed template images with orientation closest to the estimated pose angle of the tested car is selected for matching.

Table 11 Examples of database images. Each row displays car images in the same view. The first car with red boundary is the target model and the others are non-target models

<i>Target model (Toyota VIOS) and Non-target model cars in similar shape and color</i>	<i>Target model (Nissan NEW SENTRA) and Non-target model cars in similar shape and color</i>
Front ($\theta = 0^\circ$)	Front ($\theta = 0^\circ$)
	
Profile ($\theta = 45^\circ$)	Profile ($\theta = 45^\circ$)
	
Side ($\theta = 90^\circ$)	Side ($\theta = 90^\circ$)
	
Profile ($\theta = 135^\circ$)	Profile ($\theta = 135^\circ$)
	
Rear ($\theta = 180^\circ$)	Rear ($\theta = 180^\circ$)
	

All images of non-target model cars were tested first when the experiment begins. The system pre-assumed the tested cars to be the target model, and matched them against the target car template. The matching error between the tested car and the target car template was computed. Cars that were non-target models with low matching error for misclassification as the target model were called false alarms. To determine whether a tested car was target model or not, a threshold was defined to maintain the probability of false alarms as Eq. (41) at an acceptable level. For fair comparison, the probability of false alarm (P_{fa}) was maintained the same for all algorithms and the respective threshold is set thereby.

When the threshold was set properly, we matched the target model cars against the target

car template under the same P_{fa} for all algorithms. The target model cars whose matching error is smaller than the threshold are correctly classified as the target model. The rate of successful recognition (P_{rec}) of the target model cars was calculated, as illustrated in Eq. (42).

$$P_{fa} = \frac{\# \text{ of non-target model misclassified as the target model}}{\# \text{ of non-target model}} \quad (41)$$

$$P_{rec} = \frac{\# \text{ of target model correctly classified as the target model}}{\# \text{ of target model}} \quad (42)$$

6.5.2 Experimental results

The following performance indices are evaluated for the PETM, PASM, and ASMM algorithms: (1) accuracy of pose angle and center position estimation, (2) similarity of target and non-target model cars, (3) recognition rate over all orientations. Only the similarity index and recognition rate of QRAFM are measured because the QRAFM algorithm does not estimate the pose of a tested car. Finally, (4) the computation load of each functional step is demonstrated to verify the efficiency of the proposed model recognition algorithm.

6.5.2.1 Accuracy of pose angle and center position estimation

The accuracies of pose angle and center position estimations will affect the performance of ASM and mirror morphing; therefore the recognition performance is also influenced significantly. For each tested car, the ground truth of pose angle is manually defined. After the pose angle of the tested car is estimated by a pose estimation approach, either A or B, a template car that covers the estimated angle is selected. The absolute difference between the ground truth and the estimated pose is defined as the pose error without quantization (noQ). The absolute difference between the ground truth and the pose angle of the template is defined as the pose error with quantization. Because we took 10° as the angle unit for preparing the

templates, the pose error within $\pm 5^\circ$ will exist inherently for each algorithm when the tested car is matching against the selected template.

Fig. 26 and Fig. 27 show the statistical accuracies (the RMS values of the absolute differences over numerous tested car images) of the estimated pose angle and center position for the six algorithms: PETM-A, PETM-B, PASM-A, PASM-B, ASMM-A, and ASMM-B. In Fig. 26, the pose errors without quantization are also shown as PETM-A(noQ) and PETM-B(noQ). In our experiments, the quadrants of the orientations for all target model cars can be correctly identified. Due to the symmetric property, only the performances of the half orientations $[0^\circ, 180^\circ]$ are shown in the figures.

In Fig. 26, the pose errors in PETM-A are about from 3.27° to 13.02° . Larger errors appear when the orientations are within $[40^\circ, 80^\circ]$ and $[100^\circ, 150^\circ]$. The algorithm PETM-B always has smaller pose errors (from 2.50° to 6.49°) than PETM-A, especially when the orientation is within $[40^\circ, 80^\circ]$ and $[100^\circ, 150^\circ]$. This indicates that our proposed pose estimation approach (B) provides smaller and stable pose error than the conventional pose estimation approach (A) over most orientations.

However, PETM-A exhibits smaller pose errors in the neighborhood of the lateral view $[80^\circ, 100^\circ]$ than in other views. While, the PETM-B without substitution processing (denoted by PETM-B(noS) in Fig. 26) demonstrates larger pose error in these views $[80^\circ, 100^\circ]$. A hybrid strategy (with substitution processing) is employed to improve the accuracy of PETM-B in cases close to the lateral view. Therefore, PETM-B can provide accurate pose angle estimation over all orientations.

Since the pose errors of PETM-B and PETM-B(noQ) are always smaller than 7° and 5° , respectively, the proposed pose estimation approach (B) would be sufficient to support the ASM initialization, mirror morphing and template matching. PASM adopts the estimated pose angle obtained in PETM, so the pose errors of PASM-A and PASM-B are the same as those in PETM-A and PETM-B.

Fig. 27 shows the center bias of the six algorithms. Because the pose error of PETM-B is smaller than PETM-A, the center bias of PETM-B is also better than that in PETM-A. Excluding the cases of very small orientation, the center bias is from 6.12 to 8.13 pixels in PETM-B and from 6.85 to 10.00 pixels in PETM-A. Because PASM-A and PASM-B can get more location information by ASM than PETM-A and PETM-B, a more accurate center position will be re-estimated for PASM-A and PASM-B. Within the orientation $[0^\circ, 50^\circ]$ and $[140^\circ, 180^\circ]$, the center bias in PETM-A are from 7.05 to 14.01 pixels and from 6.12 to 12.00 pixels in PETM-B. Whereas, the center bias is only from 2.69 to 6.68 pixels in PASM-A and is from 1.89 to 6.58 pixels in PASM-B. Within the orientation $[50^\circ, 80^\circ]$ and $[100^\circ, 140^\circ]$, PASM still has more accurate estimation of center position, but the improvement is not so significant as in the cases of pose angle $[0^\circ, 50^\circ]$ and $[140^\circ, 180^\circ]$.

Even PASM-B remains small estimation errors of pose angle and center position, the recognition performance is still influenced dramatically by the small errors. To eliminate these errors, the ASMM algorithm adopting mirror morphing is proposed. The ASMM-A and ASMM-B algorithms normalize both the poses of the tested car and the template car to a typical (front or rear) or approximately typical (side) view. In Fig. 27, the pose errors using our proposed algorithms, ASMM-A and ASMM-B, are approximately zero when the orientation is close to 0° or 180° . When the orientation approaches to the lateral view $[80^\circ, 100^\circ]$, the quasi mirror morphing process is dominant to work for the side region, thereby increasing pose error. In ASMM-A, the maximum pose error is 4.12° , appearing at 70° and 110° . In ASMM-B, the maximum pose error is 2.54° appearing at 80° and 100° .

The errors of center position bias in the horizontal axis can also be eliminated by mirror morphing and the remaining vertical center bias in ASMM-A and ASMM-B shown is small. In Fig. 27, if excluding the cases of very small orientation, the maximum center bias is 2.76 pixels in ASMM-A and 2.23 pixels in ASMM-B. Since ASMM-A and ASMM-B have very small pose error and center bias, very good recognition performance can be obtained.

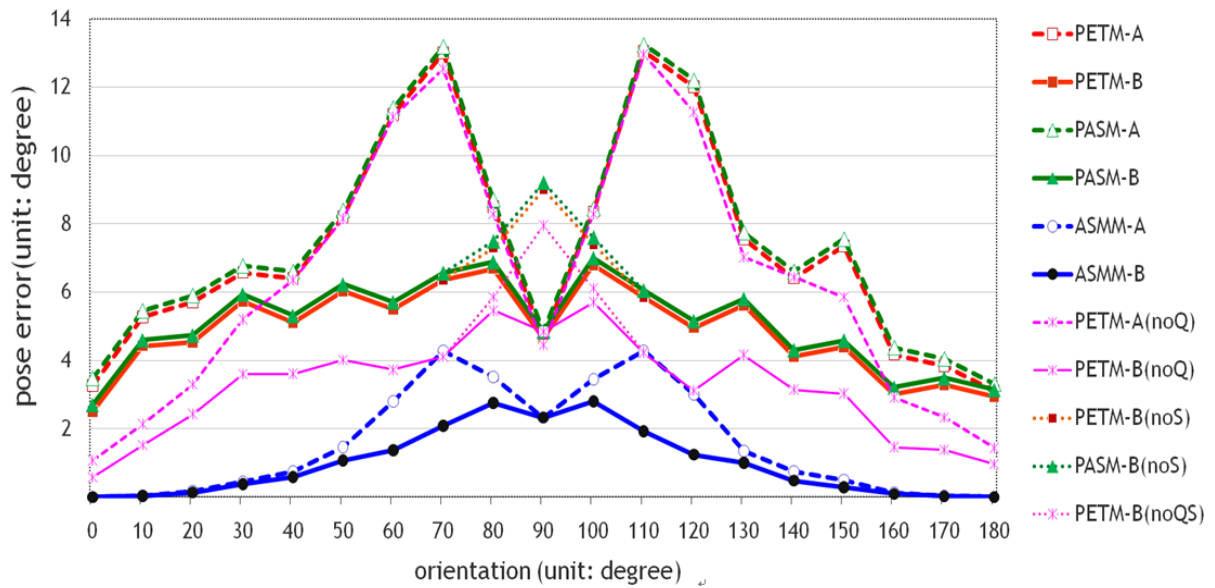


Fig. 26 Accuracy of estimated pose angle over all orientations

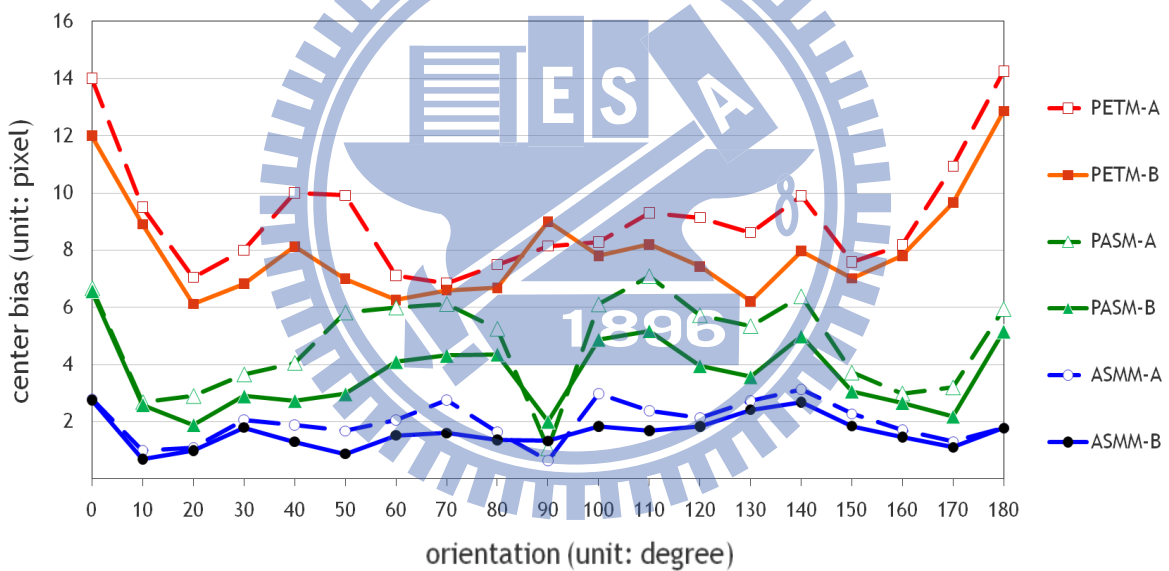


Fig. 27 Accuracy of estimated center position over all orientations

6.5.2.2 Similarity of target and non-target model car

The similarity of the target model cars and the non-target model cars determines whether they are easily to be distinguished or not. A good recognition algorithm should make the target model cars and those of the non-target model cars have very low similarity even they are in the severe pose variation condition and similar in shape and color. Fig. 28 shows the distribution of the matching errors between the tested car images to the template image.

In Fig. 28, the matching errors of the target and non-target model cars fully spread over the horizontal axis are scaled to the range of 1 to 80. The matching error is represented by the value of the symbol “i” and the equation $a \times i + b$ below the horizontal axis. The symbol “i” represents the value on the horizontal axis ($i = 1, \dots, 80$). The matching error is computed by substituting the value of “i” into the equation. For example, when the value of “i” is 26, the value of matching error in Fig. 28(a) is $2.09 \times 26 + 41.8 = 96.14$.

The red and blue curves in Fig. 28 show the distributions of the matching errors of the non-target cars and the target cars with the template car in the cases of orientation $50^\circ \pm 5^\circ$. The red curve can be served as the basis to establish a specific threshold (th_{model}) for controlling the probability of false alarm (P_{fa}). The blue curve can be utilized to evaluate the successful recognition rate (P_{rec}) under the specified threshold. Only the target model cars whose matching errors are below the threshold are correct recognitions. Therefore, the area painted by the blue color shows the cases of successful recognition, while the overlap area painted by the pink color shows the cases of fail recognition. The size of the pink area represents the similarity of target cars and non-target cars on each algorithm and the size of the blue area displays the recognition performance.

From Fig. 28(a), in the algorithm PETM-A, the pink area occupies 60.42% of the target model cars area. Thus, the recognition rate of PETM-A is 39.58%. From Fig. 28(b), in the algorithm PETM-B, the pink area occupies 52.69% of the target model cars area. Thus, the recognition rate of PETM-B is 47.31%. Our proposed pose estimation approach provides about 8% improvement from PETM-A to PETM-B.

Fig. 28 (c) and (d) show the overlapped (pink) area of target and non-target model cars of PASM-A and PASM-B, respectively. Since the algorithm PASM utilizes ASM to get more accurate center position than PETM, PASM has obviously smaller pink areas (30.21% in PASM-A and 18.75% in PASM-B) than PETM. Thus, the recognition rate of PASM-A and PASM-B would be 69.79% and 81.25%, respectively.

Due to the benefit of mirror morphing, ASMM (including ASMM-A and ASMM-B) show obviously smaller pink area than the above mentioned algorithms. In Fig. 28(e), the percentage of the pink area in ASMM-A is 13.54%, so that the recognition rate is 86.46%. In ASMM-B, because our proposed pose estimation approach (B) provides more accurate initial setting for ASM than the conventional approach (A), the shape feature of the tested car can be extracted by ASM more accurately. From Fig. 28(f), the matching error distribution of the target model cars and the non-target model cars in ASMM-B is almost completely disjoint, so that the pink area almost disappears (0%) and the recognition rate is nearly 100%.

Fig. 28(g) and (h) show the matching error distribution of the QRAFM algorithm when tested cars with orientation of $50^\circ \pm 5^\circ$ are aligned to the template car with orientation of 30° and 50° , respectively. The transformation of alignment may deform the shapes of the tested cars. The greater the pose difference between the tested cars and the template car, the more serious the deformation will be. In Fig. 28(g), because of a larger pose difference (approximately 20°) between the tested cars and the template car, the percentage of the pink area is 29.63% and the recognition rate is 70.37%. In Fig. 28(h), the absolute pose difference between the tested car and the template is smaller than 5° . Hence, the pink area occupies 10.43% of target cars; therefore, the recognition accuracy of QRAFM achieves 89.57%.

By comparing ASMM with QRAFM, the recognition rate of ASMM (Fig. 28(f)) is considerably higher than that of QRAFM (Fig. 28(g) or 28(h)). In the case of Fig. 28(h), the orientation of the tested car is close to that of the template, and ASMM is still superior to QRAFM. The QRAFM algorithm will align the tested cars including target and non-target cars to the same shape with the template; whereas, ASMM aligns the tested cars depending on their original images and mirror images. Hence, ASMM demonstrates a lower degree of similarity between the target and non-target cars than that provided by QRAFM.

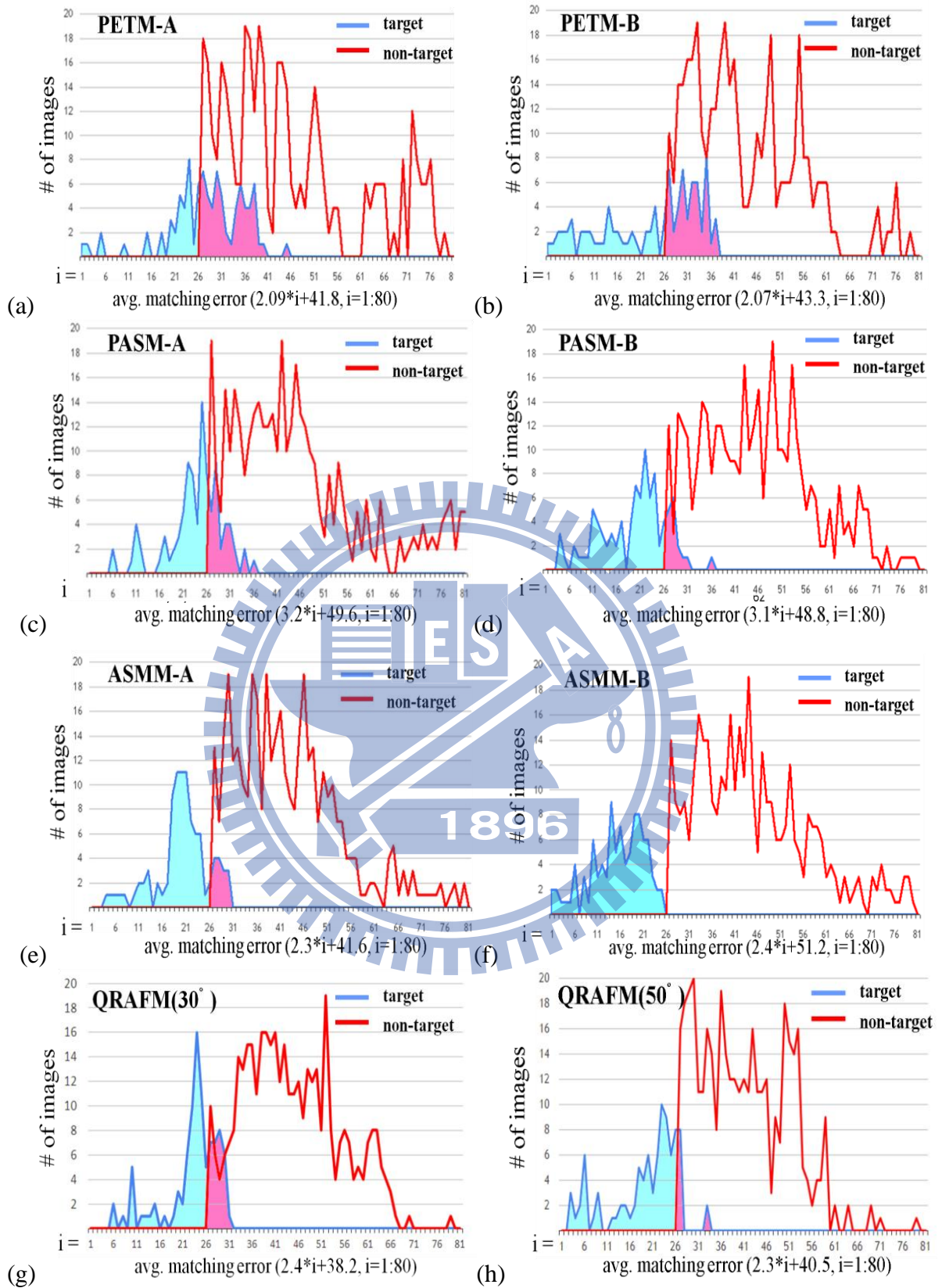


Fig. 28 Matching error distribution of (a) PETM-A, (b) PETM-B, (c) PASM-A, (d) PASM-B, (e) ASMM-A, (f) ASMM-B, (g) QRAFM adopting the template car whose orientation is 30° , and (h) QRAFM adopting 50° template on target cars and non-target cars

6.5.2.3 Recognition rate over all orientations

In Fig. 29, the algorithm PETM-A has a moderate recognition rate from 40% to 70% for the entire orientations. The recognition rate of PETM-B is improved about 5% ~ 15% than those in PETM-A when the orientation is within $[40^\circ, 70^\circ]$ and $[110^\circ, 140^\circ]$. The recognition rate of PASM-B can also be improved about 5% ~ 15% than those of PASM-A within the same orientation range. The improvement is because our proposed pose estimation approach (B) provides smaller pose error than the conventional pose estimation approach (A) when the pose angle is within $[40^\circ, 80^\circ]$ and $[100^\circ, 150^\circ]$ as shown in Fig. 26.

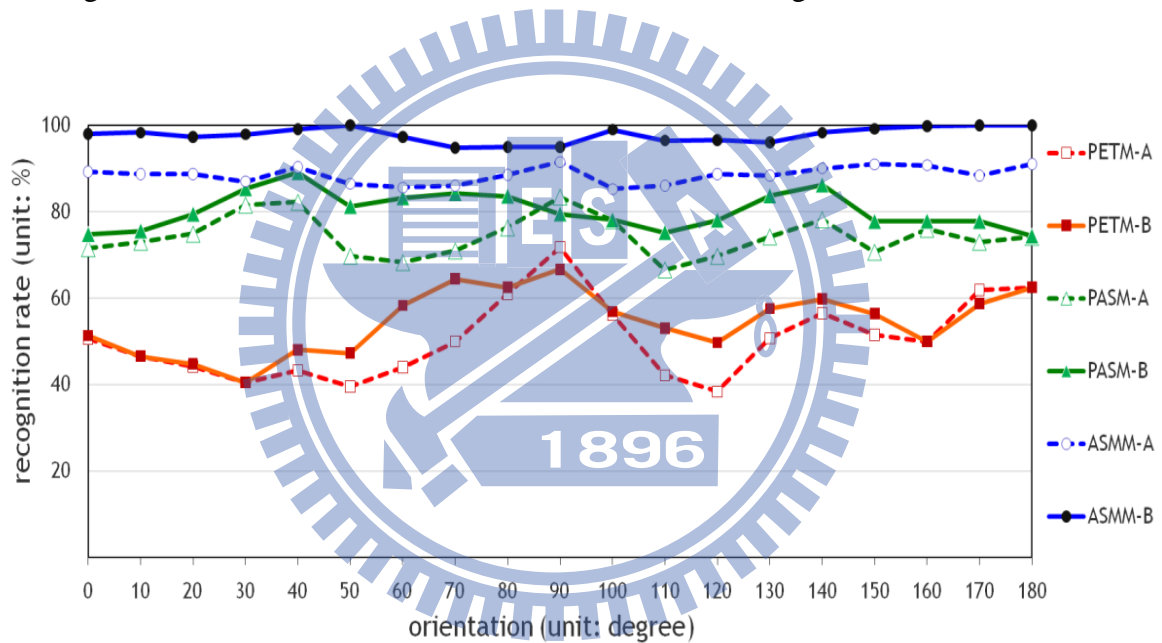


Fig. 29 Recognition rates over all orientations of the PETM, PASM and ASMM algorithms

Because the PASM algorithms (including PASM-A and PASM-B) can reduce the errors of center position significantly, the recognition performance can be highly enhanced. In Fig. 29, the recognition rate of PASM-A achieves 65% ~ 80% and the recognition rate of PASM-B achieves 75% ~ 90%. These recognition performances are much better than those of PETM-A and PETM-B and the improvement is about 10% ~ 30%.

Since mirror morphing can eliminate the estimation errors of pose angle and center position, the algorithms ASMM-A and ASMM-B can obtain very good recognition

performance. In Fig. 29, ASMM-A achieves the recognition rate from 85% to 90% and the recognition rate of ASMM-B exceeds 95% over all orientations. ASMM-B is superior to ASMM-A because our proposed pose estimation approach (B) can provide more accurate pose and center estimations. The shape features of cars will be extracted more accurately and the effect of the blurred phenomenon through mirror morphing is alleviated.

In the neighborhood of the lateral view $[80^\circ, 100^\circ]$, since the conventional pose estimation approach has smaller pose error in this range of views, the algorithms denoted by A (PETM-A, PASM-A, ASMM-A) demonstrate superior recognition rates than those in the other views. By the hybrid strategy, the algorithms denoted by B (PETM-B, PASM-B, ASMM-B) have approximately the same pose error and recognition rate in all orientations. And, because the tolerance of pose error is greater in cases close to lateral view, the quasi mirror morphing process which we design for side region can still work well in these cases. As shown in Fig. 29, the proposed algorithms (ASMM-A and ASMM-B) still have high recognition rates in the neighborhood of the lateral view. In ASMM-B, a recognition rate exceeding 95% can be achieved in all orientations, even in the cases close to the lateral view.

Fig. 30 compares the recognition rate of ASMM-B (blue curve) and QRAFM with 45° and 135° templates (purple curve) over all orientations. The recognition rate of QRAFM decreases in conjunction with the increment of pose-difference between the template and the tested cars. For the QRAFM algorithm with 45° template, the recognition rate achieves over 80% when the tested car orientation is within $[40^\circ, 50^\circ]$. If the tested car orientation falls in the range of $30^\circ \pm 5^\circ$ or $60^\circ \pm 5^\circ$, the recognition rate decreases to smaller than 70%. Moreover, the recognition rate is lower than 50% when the orientation is smaller than 20° or larger than 70° . ASMM obviously outperforms QRAFM.

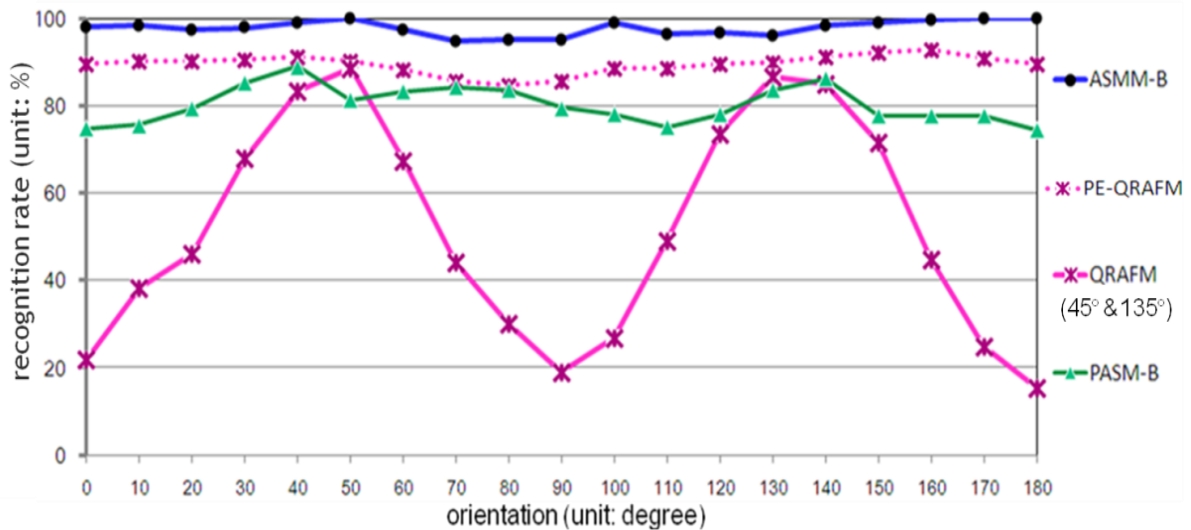


Fig. 30 Recognition rates over all orientations of the ASMM-B, PE-QRAFM , QRAFM with 45 ° and 135 ° templates, and PASM-B algorithms

Since the QRAFM algorithm exhibits superior performance when the pose difference between the template and the tested car is limited, we combine our proposed pose estimation approach with QRAFM and compare the integrated algorithm with ASMM. The integrated algorithm is termed PE-QRAFM (Pose Estimation – Quasi Rigid Alignment and Flexible Matching). The recognition rate of PE-QRAFM is shown as the dashed purple curve in Fig. 30. The recognition rate of PE-QRAFM is improved averaging to approximately 90%; however, our proposed algorithm ASMM achieves a superior recognition rate averaging around 98%. Therefore, although the QRAFM algorithm is assisted by the proposed pose estimation approach, our proposed ASMM algorithm which utilizes the mirror morphing technique is superior to the QRAFM algorithm for car model recognition, especially in the cases of cars with similar shape and color.

6.5.2.4 Computation load

Our proposed car model recognition algorithm was implemented on an AMD Athlon II X2 240, 2.0 GB RAM, 2.81GHz PC with Language C environment. The average computation time of each step performed on a tested image (image size = 150×500 pixels) is reported in

Table 12. The computation time for the whole algorithm is about 0.530 sec and such computation time is applicable to most real systems.

Table 12 Average computation time of each step in the car model recognition system

<i>Step</i>	(1) <i>Pose & Center Estimation</i>	(2) <i>Shape Feature Extraction</i>	(3) <i>Car Region Segmentation</i>	(4) <i>Mirror Morphing</i>	(5) <i>Template Matching</i>	<i>Whole Algorithm</i>
<i>Computation time (sec)</i>	0.072	0.140	0.026	0.283	0.009	0.530



Chapter 7 Conclusions and Future Works

7.1 Conclusions

This dissertation proposes a traffic monitoring system to determine colors, classify sizes, and recognize models of cars effectively and efficiently. A tri-states architecture mechanism for bright, dark, and colored cars is proposed. The mechanism includes an SARM algorithm to effectively extract the car body and classify the vehicle color in challenging cases with unknown car type, unknown viewpoint, and non-homogeneous light reflection conditions. According to the experimental results, the tri-states mechanism obtained satisfactory car body covering precision and recall for three categories of car images. Satisfactory classification accuracies of color category and color type can also be achieved. The entire computation time of the tri-states mechanism is approximately 0.057 sec per image. This length of computation time is applicable to most real systems for car body extraction and color classification.

Vehicle pose and size are also important appearance features of cars. In the system, the symmetric center is searched and utilized to estimate the head width of the vehicle. Two ratios are defined according to the head (or rear) width for pose estimation and size classification, respectively. Compared with the traditional approaches based on the ratio of car width to car height, the new ratios are used to provide better pose estimation accuracy under small to large orientations and better size classification accuracy under small and medium orientations. Under large orientations for car size classification, a method detecting the concavity triangles is proposed to separate small and big cars. According to the experimental results, the proposed method provides satisfactory car pose estimation and size classification accuracy under multiple views and the computation time is acceptable for real time traffic monitoring systems.

Because cars in the same class tend to appear similar and the shape features of vehicles vary significantly from those of the template under different poses, recognizing car features remains a challenge nowadays. Pose and center estimations can be performed to limit the influences of the pose variation. However, very accurate pose and center estimation is difficult to achieve in practice and even a little deviation in the estimated pose and center would degrade the recognition performance significantly. In this dissertation, we present a mirror morphing scheme which utilizes the symmetric property of cars to eliminate the pose variation and center bias; therefore, the pose problem can be accommodated. According to the experimental results, the similarity of the target model cars and non-target model cars through the mirror morphing scheme becomes smaller than that without mirror morphing. Consequently, it is easier to distinguish the target model cars from a lot of non-target model cars. Therefore, our proposed car model recognition system can achieve a very good recognition rate with very low probability of false alarm over all orientations in the cases of the cars with similar shapes and colors.

7.2 Future works

Our proposed car model recognition system performs well for car images captured from the ground level or a slightly higher view. In many practical situations such as holding a camera in hands, mounting a camera on a tripod, or placing a camera on a driving car, the proposed approach achieves satisfactory recognition accuracy for car models. In surveillance systems, cameras are generally deployed at fixed locations with fixed heights, so the vertical view angle is limited to within a small range. In this situation, the proposed approach can still improve the car model recognition rate. In more challenging situations where cameras are set up in locations with various vertical view angles, a more intelligent pose estimation algorithm should be developed and more templates covering a wide range of vertical view angles should be prepared. Cases involving a variety of vertical view angles will be dealt with in the future.

The proposed car model recognition system is designed for cars exhibiting symmetry in the front and rear regions. It can recognize sedans, station wagons, vans, etc., well in all orientations. A number of vehicle types, such as SUVs, Jeeps, and trucks, have the symmetric properties in the front region, but usually lack symmetry in the rear region. Therefore, our proposed system is only able to recognize such vehicles in a limited number of orientations, in which the front region of the tested car is visible. In the future, a more intelligent mechanism which is able to handle the cars without symmetric properties in the rear region will be designed. If a car is identified that its symmetric property is not obvious, mirror morphing is not performed to prevent the recognition accuracy from being destroyed.

The current version of the proposed car model recognition approach only considers the scenario that the cars with the same shape but different colors are different car models. To deal with the more challenging scenario that cars with the same shape but different colors are considered as the same model, an integrated system which combines the color determination and car model recognition algorithms is going to be presented. The car color determination algorithm extracts the car body and classifies the vehicle color first. Then, a repaint procedure translates the pixels on the extracted car body of a tested car into the same color of the template car. Finally, the car model recognition algorithm is employed to recognize the car model. The performance improvement benefited from integrating the repaint procedure into the car color determination and car model recognition is under evaluation.

License plate characters are not considered in this research since they are easily hidden or counterfeited. If the appearance features of vehicles can be correctly identified, the sizes and slopes of characters on license plates can be used for improving the accuracy of pose estimation of vehicles, moreover, the car model recognition performance. Therefore, integrating the information of license plate into our proposed car model recognition system is also the future work of this research.

Reference

- [1] B. Ran and H. X. Liu, "Development of a Vision-Based Vehicle Detection and Recognition System for Intelligent Vehicles" Transportation Research Board Annual Meeting, pp. 130-138, 1999
- [2] S. L. Chang, L. S. Chen, Y. C. Chung, and S. W. Chen, "Automatic License Plate Recognition", IEEE Transactions on Intelligent Transportation Systems, vol. 5, no. 1, pp. 42-53, 2004
- [3] C. E. Anagnostopoulos, I. E. Anagnostopoulos, I. D. Psoroulas, V. Loumos, and E. Kayafas, "License Plate Recognition from Still Images and Video Sequences: A Survey," IEEE Transactions on Intelligent Transportation System, vol. 9, no. 3, pp. 377-391, 2008
- [4] K. J. Kim, S. M. Park, and Y. J. Choi, "Deciding the Number of Color Histogram Bins for Vehicle Color Recognition," IEEE Asia-Pacific Services Computing Conference, pp. 134-138, 2008
- [5] X. Li, G. Zhang, J. Fang, J. Wu, and Z. Cui, "Vehicle Color Recognition Using Vector Matching of Template," International Symposium on Electronic Commerce and Security, pp. 189-193, 2010
- [6] L. M. Brown, "Example-Based Color Vehicle Retrieval for Surveillance," International Conference on Advanced Video and Signal Based Surveillance, pp. 91-96, 2010
- [7] N. Baek, S. M. Park, K. J. Kim, and S. B. Park, "Vehicle Color Classification Based on the Support Vector Machine Method," Communications in Computer and Information Science, vol. 2, part 24, pp. 1133-1139, 2007
- [8] J. W. Son, S. B. Park, and K. J. Kim, "A Convolution Kernel Method for Color Recognition," International Conference on Advanced Language Processing and Web Information Technology, pp. 242-247, 2007

- [9] M. Butzke, A. G. Silva, M. S. Hounsell, and M. A. Pillon, "Automatic Recognition of Vehicle Attributes – Color Classification and Logo Segmentation", HIFEN, vol. 32, no. 62, 2008
- [10] D. Santos and P. L. Correia, "Car Recognition Based on Back Lights and Rear View Features", Workshop on Image Analysis for Multimedia Interactive Services, pp. 137-140, 2009
- [11] Y. T. Wu, J. H. Kao, and M. Y. Shih, "A Vehicle Color Classification Method for Video Surveillance System Concerning Model-Based Background Subtraction," Advances in Multimedia Information Processing – PCM Lecture Notes in Computer Science, vol. 6297, pp. 369-380, 2010
- [12] A. Elgammal, D. Harwood, and L. Davis, "Non-parametric Model for Background Subtraction," 6th European Conference on Computer Vision, Lecture Notes in Computer Science, vol. 1843, pp. 751-767, 2000
- [13] Z. A. Aghbari and R. O. Al-Haj, "Hill-manipulation: An Effective Algorithm for Color Image Segmentation," Image and Vision Computing, vol. 24, no. 8, pp. 894-903, 2006
- [14] T. Ohashi, Z. Aghbari, and A. Makinouchi, "Hill-climbing Algorithm for Efficient Color-Based Image Segmentation," IASTED International Conference on Signal Processing, Pattern Recognition, and Applications, pp. 17-22, 2003
- [15] R. T. Tan and K. Ikeuchi, "Separating Reflection Components of Textured Surfaces Using a Single Image," IEEE International Conference on Computer Vision, vol. 2, pp. 870-877, 2003
- [16] D. Santos and P. L. Correia, "Car Recognition Based on Back Lights and Rear View Features", Workshop on Image Analysis for Multimedia Interactive Services, pp. 137-140, 2009

- [17] X. Wen, H. Zhao, N. Wang and H. Yuan, "A Rear-Vehicle Detection System for Static Images Based on Monocular Vision", International Conference on Control, Automation, Robotics and Vision, pp. 1-4, 2006
- [18] T. Hou, S. Wang, and H. Qin, "Vehicle Matching and Recognition under Large Variations of Pose and Illumination," International Conference on Computer Vision and Pattern Recognition, pp. 24-29, 2009
- [19] B. Barrois, S. Hristova, C. Wohler, F. Kummert, and C. Hermes, "3D Pose Estimation of Vehicles Using a Stereo Camera," IEEE Intelligent Vehicles Symposium, pp. 267-272, 2009
- [20] S. Nedevschi, R. Danescu, T. Marita, F. Oniga, C. Pocol, S. Sobol, C. Tomiuc, C. Vancea, M. M. Meinecke, T. Graf, T. b. To, and M. A. Obojski, "A Sensor for Urban Driving Assistance Systems Based on Dense Stereovision," IEEE Intelligent Vehicles Symposium, pp. 276-283, 2007
- [21] A. Collet, D. Berenson, S. S. Srinivasa, and D. Ferguson, "Object Recognition and Full Pose Estimation from a Single Image for Robotic Manipulation" IEEE International Conference on Robotics and Automation, pp. 48-55, 2009
- [22] Y. Shan, H. S. Sawhney, R. Kumar, "Unsupervised Learning of Discriminative Edge Measures for Vehicle Matching between Nonoverlapping Cameras," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 30, no. 4, pp. 700-711, 2008
- [23] Y. Guo, C. Rao, S. Samarasekera, J. Kim, R. Kumar, and H. Sawhney, "Matching Vehicles under Large Pose Transformations using Approximate 3D Models and Piecewise MRF Model," IEEE Conference on Computer Vision and Pattern Recognition, pp. 1-8, 2008
- [24] S. Savarese and F. F. Li, "3D Generic Object Categorization, Localization, and Pose Estimation," International Conference on Computer Vision, pp. 1-8, 2007

- [25] T. Hou, S. Wang, and H. Qin, "3D Model-Driven Vehicle Matching and Recognition," *Machine Vision Beyond Visible Spectrum, Augmented Vision and Reality*, vol. 1, pp. 65-85, 2011
- [26] J. W. Hsieh, S. H. Yu, Y. S. Chen, and W. F. Hu, "Automatic Traffic Surveillance System for Vehicle Tracking and Classification," *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, no. 2, pp. 175-187, 2006
- [27] H. Y. Chang, C. M. Fu, and C. L. Huang, "Real-time Vision-based Preceding Vehicle Tracking and Recognition," *IEEE Intelligent Vehicles Symposium*, pp. 514-519, 2005
- [28] X. Ma and W. Eric L. Grimson, "Edge-based Rich Representation for Vehicle Classification," *International Conference on Computer Vision*, vol. 2, pp. 1185-1192, 2005
- [29] Z. Chen, N. Pears, M. Freeman, and J. Austin, "Road Vehicle Classification using Support Vector Machines," *IEEE Conference on Intelligent Computing and Intelligent Systems*, pp. vol. 4, 214-218, 2009
- [30] S. Y. Chien, S. Y. Ma, and L. G. Chen, "Efficient Moving Object Segmentation Algorithm Using Background Registration Technique," *IEEE Transactions on Circuits and Systems for Video Technology*, vol.12, no.7, pp. 577-586, 2002
- [31] B. S. Chen, Y. Q. Lei, and W. W. Li, "A Novel Background Model for Real-Time Vehicle Detection," *IEEE International Conference on Signal Processing*, vol. 2, pp. 1276-1279, 2004
- [32] S. C. Liu, C. W. Fu, S. Chang, "Statistical Change Detection with Moment under Time-Varying Illumination," *IEEE Transactions on Image Processing*, vol. 7, no. 9, pp. 1258-1268, 1998
- [33] D. S. Zhang and G. Lu, "Segmentation of Moving Objects in Image Sequence: A Review," *Circuits, Systems, and Signal Processing*, vol. 20, no. 2, pp. 143-183, 2001

- [34] A. H. S. Lai, G. S. K. Fung, and N. H. C. Yung, "Vehicle Type Classification from Visual Based Dimension Estimation" IEEE International System Conference, pp. 201-206, 2001
- [35] Y. Guo, S. Hsu, H. S. Sawhney, R. Kumar, and Y. Shan, "Robust Object Matching for Persistent Tracking with Heterogeneous Features," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 29, no. 5, pp. 824-839, 2007
- [36] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham, "Active Shape Models – Their Training and Application" Computer Vision and Image Understanding, vol. 61, no. 1, pp. 38-59, 1995
- [37] T. F. Cootes, G. J. Edwards, and C. J. Taylor, "Active Appearance Models" IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 23, no. 6, pp. 681-685, 2001
- [38] M. Hutter and N. Brewer, "Matching 2-D Ellipses to 3-D Circles with Application to Vehicle Pose Identification," International Conference on Image and Vision Computing New Zealand, pp. 153-158, 2009
- [39] J. Prokaj and G. Medioni, "3D Model Based Vehicle Recognition," Workshop on Applications of Computer Vision, pp. 1-7, 2009
- [40] A. Kuehnle, "Symmetry-based Recognition of Vehicle Rears," Pattern Recognition Letters, vol. 12, no. 4, pp. 249-258, 1991
- [41] D. S. Kim and S. I. Chien, "Automatic Car License Plate Extraction Using Modified Generalized Symmetry Transform and Image Warping," IEEE International Symposium on Industrial Electronics, vol.3, pp. 2022-2027, 2001
- [42] U. Kadow, G. Schneider, and A. Vukotich, "Radar-Vision Based Vehicle Recognition with Evolutionary Optimized and Boosted Features," IEEE Intelligent Vehicles Symposium, pp. 13-15, 2007

- [43] S. Wan, S. Kodagoda, and S. Sehestedt, "Multiple Cue Based Vehicle Detection and Tracking for Road Safety," International Conference on Intelligent Technologies, pp. 12-14, 2007
- [44] H. Z. Gu and S. Y. Lee, "A View-Invariant and Anti-Reflection Algorithm for Car Body Extraction and Color Classification", Multimedia Tools and Applications, 2012, DOI: 10.1007/s11042-012-0996-1
- [45] R. S. Hunter, "Accuracy, Precision, and Stability of New Photo-electric Color Difference Meter," Journal of the Optical Society of America, vol.38, no.7, pp.661, 1948
- [46] "Colormap by MathWorks",
URL: <http://www.mathworks.com/help/techdoc/ref/colormap.html>
- [47] G. D. Finlayson and G. Schaefer, "Solving for Color Constancy using a Constrained Dischromatic Reflection Model," International Journal of Computer Vision, vol. 42, no. 3, pp. 127-144, 2001
- [48] S. Lin, Y. Li, S. B. Kang, X. Tong, and H. Y. Shum, "Diffuse-specular Separation and Depth Recovery from Image Sequences," Europe Conference of Computer Vision, pp. 210-224, 2002
- [49] R. M. Haralick and L. G. Shapiro, Computer and Robot Vision, Volume I, Addison-Wesley, pp. 28-48, 1992
- [50] S. Kim, S. Park, and M. Kim, "Central Object Extraction for Object-Based Image Retrieval," International Conference on Image and Video Retrieval, pp. 39-49, 2003
- [51] S. Sural, G. Zian, and S. Pramanik, "Segmentation and Histogram Generation Using the HSV Color Space for Image Retrieval," International Conference on Image Processing, vol.2, pp. 589-592, 2002
- [52] S. Theodoridis and K. Koutroumbas, "Pattern Recognition" 4th edn. Academic Press, ISBN: 1597492728, pp. 119-128, 2008

- [53] C. C. Chang and C. J. Lin, "Training nu-Support Vector Classifiers: Theory and Algorithms," *Neural Computation*, vol. 13, no. 9, pp. 2119-2147, 2001
- [54] "Dataset of the car model recognition system and the tri-states system",
URL: <http://www.cs.nctu.edu.tw/~hcku/>
- [55] M. Ozuysal, V. Lepetit, and P. Fua, "Pose Estimation for Category Specific Multiview Object Localization," *International Conference on Computer Vision and Pattern Recognition*, pp. 778-785, 2009
- [56] H. Z. Gu and S. Y. Lee, "Car Model Recognition by Utilizing Symmetric Property to Overcome Severe Pose Variation," *Machine Vision and Applications*, 2012, DOI: 10.1007/s00138-012-0414-8
- [57] G. Wolberg, "Image Morphing: A Survey," *Visual Computer*, vol. 14, pp. 360-372, 1998
- [58] T. F. Cootes, G. J. Page, C. B. Jackson, C. J. Taylor, "Statistical Grey-Level Models for Object Location and Identification," *Image and Vision Computing*, vol. 14 no. 8, pp. 533-540, 1996
- [59] T. Beier, and S. Neely, "Feature-based Image Metamorphosis," *SIGGRAPH*, pp. 35-42, 1992
- [60] R. Brunelli, "Template Matching Techniques in Computer Vision: Theory and Practice," John Wiley & Sons, New York, 2009