# Robust Curved Road Boundary Identification Using Hierarchical Clustering

by

Nurjahan Parvin

A thesis submitted in conformity with the requirements
for the degree of Master of Computer Science
Faculty of Science (Computer Science)
University of Ontario Institute of Technology

Supervisor(s): Dr. Ken Q. Pu and Dr. Faisal Z. Qureshi

# Abstract

Robust Curved Road Boundary Identification Using Hierarchical Clustering

Nurjahan Parvin

Master of Computer Science

Faculty of Science

University of Ontario Institute of Technology

2013

We develop a new method for automatic curved road boundary detection in images captured by traffic cameras. The proposed method combines data driven (edge segments) and model based (2nd degree polynomial models for road boundaries) techniques to identify dominant road boundary in images exhibiting extreme weather conditions, low visibility and poor lighting. The proposed method constructs a ranked list of possible road boundaries through agglomerative hierarchical clustering of edge segments. Each node in the hierarchical clustering is a potential road boundary. Top ranked road boundaries are paired with each other to identify potential road regions. The road regions are then ranked using appearance and perspective cues and the top ranked road region is used to construct the dominant road boundary in the image. We evaluate our method on a realistic dataset captured by traffic cameras managed by Ontario's Ministry of Transportation.

# Acknowledgements

I would like to express my sincere gratitude to my supervisors, Dr. Ken Q. Pu and Dr. Faisal Z. Qureshi for their continuous guidance, support, motivation, and patience during my graduate studies. Special thanks to my fellow lab members, Mohamed Helala, Luis Zarrabeita, Zheng Wang, Wiktor Starzyk and Jordan Stadler, for their helpful suggestions and encouragement. They made graduate experience exciting and rewarding.

Finally, I wish to thank to my beloved family, for their understanding and endless love and care.

# Contents

# List of Tables

# List of Figures

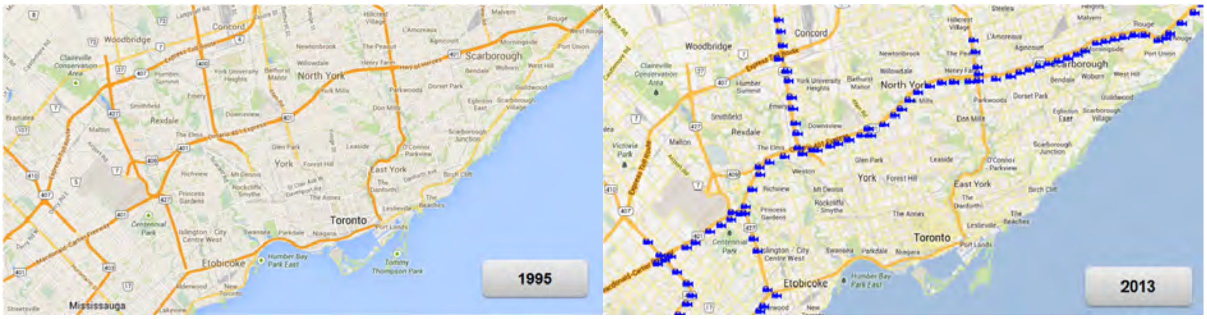
# Chapter 1

# Introduction

Traffic surveillance systems are now pervasive. Cameras are being deployed along roads and highways in increasing numbers. For example, Figure 1.1 (a) illustrates the number of cameras installed on a section of Highway 401 (that passes through Toronto, Canada) in the year 1995; Figure 1.1 (b), on the other hand, depicts the same road section in the year 2013. Notice that this section of the road alone has roughly 200 cameras in the year 2013.[1] Video collected by these traffic cameras is typically sent to a central monitoring station for subsequent processing, monitoring and storage. The sheer volume of video data collected by these systems suggest that we need automatic techniques to analyze this data with a view to identify situations that need our attention, say a stranded vehicle.

The ability to analyze video data from traffic cameras has many applications, such as traffic reporting, responding to emergencies, toll collection, improved road safety, etc. [2, 1]. The capability to automatically detect road boundaries can help in subsequent analysis of the video data captured by traffic cameras. This work develops a technique for detecting dominant road boundary from a single image captured by a traffic surveillance camera. A unique aspect of our work is its ability to detect road boundaries in images that exhibit a range of environmental and lighting effects. For example, images captured during a Canadian snow storm or an image

[1] The video feed captured from these cameras is publicly available and our data set consists of images captured by these cameras.



(a) (b)

**Figure 1.1** This figure depicts the number of traffic cameras installed on a section of Hwy 401 that passes through the Greater Toronto Area in the years 1995 (a) and 2013 (b). Notice that there were no traffic cameras installed on this section of the highway in the year 1995.

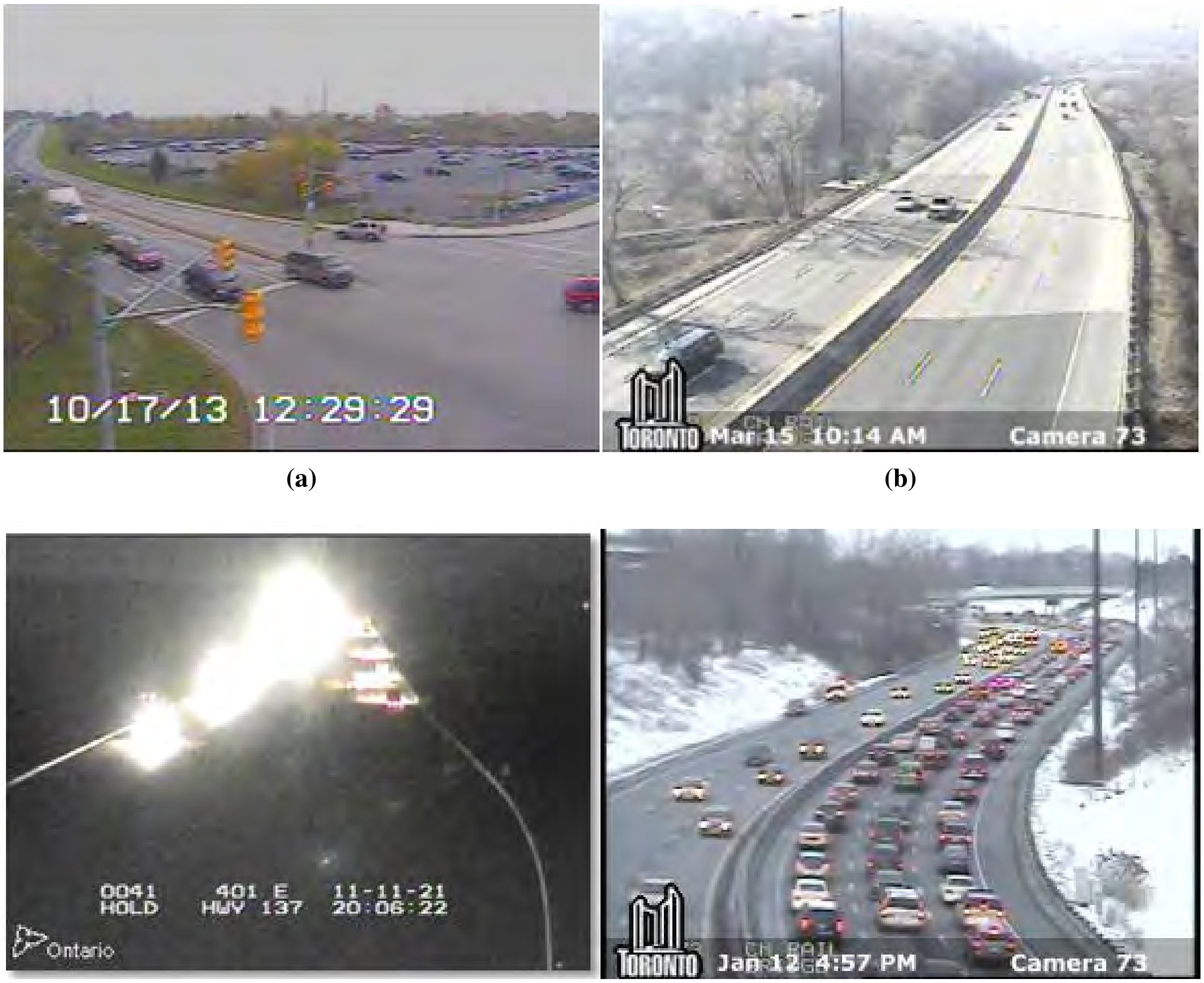
captured at night. Additionally, the proposed technique works well on low-resolution images exhibiting compression artifacts. Our use of low-resolution imagery for the problem of road boundary identification is motivated by the fact that bandwidth is at a premium in these traffic surveillance systems. Indeed bandwidth consumption is directly correlated with the energy expenditure at each camera and can determine the overall lifespan of these cameras.

## 1.1 Unique challenges

Identifying road boundaries in images captured by traffic cameras is challenging as images captured by these cameras exhibit the following four characteristics (Figure 1.2).

### Low resolution imagery

As previously mentioned our algorithm works with low-resolution imagery (say $320 \times 240$ pixels). The reason for this is simple. There are a lot of traffic cameras, each of which is continuously capturing video data at roughly 30 frames per second. One way to deal with this volume of data is to 1) down-sample it (both spatially and temporally) and 2) compress it. This observation led us to focus on algorithms that do not require high-resolution (say, 720p or 1080p) images. The algorithm also does not require multiple images (i.e., a video clip captured by a traffic camera) to identify road boundaries.



(a) (b)

(c) (d)

**Figure 1.2** A subset of the images from our data set.

### Extreme environmental factors

Being outdoor imagery, we have little control over the environmental conditions under which these images are captured. Consequently these images exhibit every environmental condition imaginable, from rain and fog to snow and blinding sunlight. An algorithm must be able to correctly identify road boundaries in the presence of extreme environmental conditions observed in these images.

### Challenging lighting conditions

The images captured by traffic surveillance cameras also exhibit a range of lighting conditions. A road captured during day time, for example, looks very different when viewed at night. Images captured at night time, especially, suffer from low visibility and low contrast, rendering appearance based methods unsuitable. We noticed that in a number of images captured at night, the roads are only visible due to headlights of the passing vehicles.

### Unknown road boundary geometry

The geometry of the road boundaries is not known *a priori*, suggesting that any proposed algorithm must be able to handle a variety of road boundaries. Presently our algorithm assumes that the road boundaries can be represented as quadratic polynomials. This assumption allows us to detect curved roads, however, the proposed algorithm is currently not able to identify S-shaped boundaries.

## 1.2 Contributions

This thesis makes the following contributions:

**1:** We have developed a novel technique for identifying curved road boundaries from a single traffic camera image under a variety of extreme weather conditions and challenging lighting conditions.

**2:** We have developed a new distance metric that appears suitable for performing hierarchical bottom-up clustering of edge segments (data-drive, bottom-up) to construct (model-drive, top-down) road boundary hypotheses. Each road boundary hypothesis is ranked using this metric and the top ranked boundaries are paired with each other to construct hypotheses for road regions. We have also developed an automated scheme for ranking road regions based upon probabilistic appearance differences between road and non-road regions and perspective cues.

**3:** We have evaluated our method on a realistic data set captured by traffic cameras installed along the section of Hwy 401 that passes through the Greater Toronto Area. These traffic cameras are managed by Ontario's Ministry of Transportation.

## 1.3 Overview

The remainder of this thesis is organized as follows. Chapter 2 reviews related work. We develop our method in the following chapter and provide evaluation results in Chapter 4. We conclude this thesis with a brief discussion and a list of possible directions for future work in Chapter 5.

# Chapter 2

# Background Study

As mentioned in the previous chapter, we wish to develop algorithms that output curved road boundaries automatically with no human input. The techniques for road boundary and lane detection can be broadly divided into three categories: (1) activity-based (Stewart *et al.* [33] and Melo *et al.* [24]), (2) feature driven (Soumelidis *et al.* [32], Aly [4], and Kong *et al.* [16]), and (3) model driven (Zhou *et al.* [39] and Wang *et al.* [35]). The activity-based techniques separate the traffic scene into active (road) and inactive regions (non-road) and generate an activity map based on vehicles motion information. While the feature based techniques extract low level image features like color or edges in order to estimate the road boundary, the model based techniques fit a linear or polynomial equation to the extracted features for representing lane or road boundary.

Our work presents an algorithm, a combination of feature (brightness, color, and texture) and model (as curves in 2D space described by $2^{nd}$ degree polynomial equation) based approaches focusing on satisfying all the properties we mentioned to estimate dominant road boundaries. A single traffic image is analysed here for this purpose.

| **References** | **Feature type** | **Goal** | **Requirements** | **Limitations** | **Our work** |
|---|---|---|---|---|---|
| [16,22,27,28] | | Lane detection | Need lane markings to be visible; works on a single image | Fails if lane markings are not visible; easily affected by the occlusions caused by vehicular traffic | Can work with lane markings |
| [11,21] | Color based | Road boundary detection | Works on a single image | Perform poorly under extreme environmental and challenging lighting conditions | Performs well under extreme environmental and challenging lighting conditions |
| [1,14,24] | Texture based | Road boundary detection | Works on a single image | Cannot work on curved roads | Able to handle curved roads |

**Table 2.1** Comparison between existing feature based approaches and the proposed approach.

| **References** | **Goal** | **Requirements** | **Limitations** | **Model** | **Our work** |
|---|---|---|---|---|---|
| [4,35] | Road and lane detection | Video; road markings must be visible | Cannot handle curved roads | 1st degree polynomial | Does not need lane markings to be visible |
| [14,23,24] | Road boundary detection | Needs single image; assumes that road boundaries meet at the vanishing point | Cannot handle curved roads | 1st degree polynomial | Able to detect curved road boundaries |
| [9,19,34] | Multiple lane detection | Needs lane markings to be visible | Affected by occlusions due to vehicular traffic | Parabolic model | Able to handle occlusions due to vehicular traffic |
| [12,28,31] | Lane detection | Needs lane markings to be visible | Requires manual selection of control points | Splines | Does not require the specification of control points |

**Table 2.2** Comparison between existing model based approaches and the proposed approach.

| References | Goal | Requirements | Limitations | Our work |
|---|---|---|---|---|
| [20] | Lane detection | Video; assumptions about lane widths | Performs poorly on curved roads | Can handle both straight and curved roads |
| [6,17] | Lane detection | Video | Requires somewhat accurate tracking of vehicles present on the road | Does not need to track vehicles present in the road; works on a single image |
| [8] | Road boundary detection | Video | Cannot handle curved roads | Can handle both straight and curved roads; works on a single image |

**Table 2.3** Comparison between existing activity based approaches and the proposed approach.

## 2.1 Basic road detection techniques

### 2.1.1 Feature driven

Low level image features are extracted from the traffic image to identify lane/road boundaries. For road detection, typically brightness, color, and texture features are used. Moreover, lane marking is searched as the evidence for lane detection.

In most of the roads, lane boundaries are visible by different types of lane markings [23]. The shape and color of those lane markings vary from continuous lines through dashed lines and from white to yellow colors respectively. Therefore, the image features and appearance of lane and road surfaces are different and can be used as cues to detect lane boundaries. The idea of gradient based features and their variants originate based on above hypothesis. Methods relying on gradient has been applied by Samadzadegan *et al.* [30], Sawano and Okada [31], and Nieto *et al.* [26] to extract various features.

Brightness difference on road surface is another important feature to locate lane boundary. Several authors (Huang *et al.* [12], Labayrade *et al.* [19], and Wu *et al.* [37]) presented algorithms involving scanning the image row by row to search low-high-low intensity pattern with the assumption of the narrow shape of lane markings and having brighter intensity than their surrounding area. Huang *et al.* [12] and Wu *et al.* [37] suggested to apply a *box filter* to the image. Labayrade *et al.*, on the other hand, proposed to use a *step filter* to convolve the image and then searched for pairs of adjacent responses with opposite signs [19].

The approaches described above fail on roads which don't have man-made lane markings. In that case, different cues are proposed to recognize the road boundaries based on the type of road and environment. Few approaches include curbs especially for urban environments, barriers which are seen on highway roads, and even dirt roads where only color or textural difference between road and off-road areas can specify the boundaries. Each of the cues is suitable for a specific system and no single cue can handle all the cases. Selection of those cues varies depending on the hypothesis that one makes and the types of road to handle.

The work proposed by Katramados *et al.* [14] is based on the idea of using histogram for the description of road color distribution. It first converts the image color into several illumination-invariant color channels. Then, it defines a safe window at the bottom area of the image. Pixels of that area are placed in the histograms based on their channel intensities. Next, likelihood of the intensity for the rest of the image is computed. Finally, voting is performed to segment the image. Nefian and Bradski [25] relied on similar types of approach considering a *Gaussian mixture model* instead of histogram. It also computes safe window and color likelihood like [14] in addition of integration with a *Bayesian network* [10] framework. Here, each pixel is a node and edges represent the pixels adjacency. This work can also handle the shadow affect by adding some hidden variables to the networks node. *Bayesian network* framework segments each image row into three parts called off-road, on-road, and then off-road again to separate the road region from outside.

There has been significant amount of work that deals with texture features to examine road boundaries [28, 17, 3]. Although texture differs mostly as the material of the road changes, however the footprints made by wheels of preceding vehicles add a strong directional component to the texture. In straight road, the assumption can be made that all directional components will meet a common vanishing point. *Gabor filter* is applied by Rasmussen and Korah [28], Kong *et al.* [17] to determine the texture's dominant direction on each image point. Next, voting is performed to predict the vanishing point. It then searches for the two most extrinsic rays that pass though the vanishing point and have high enough directional support. Finally, the road region is declared as the inner area between those two rays. Similar technique was used in the work proposed by Alon *et al.* [3] considering alternative selection of textural features. A variety of choices are examined and they find that applying *walsh-hadamard* [21] features decrease computational overhead dramatically with maintaining similar level of system accuracy.

### 2.1.2 Model driven

Generally roads and lanes follow some geometric structures such as straight lines or a variety of curves etc. Therefore, boundary detection can be guided by fitting a geometric model to the extracted features in a particular frame. In most cases, the model has to operate with missing data and a large amount of outliers.

In general, the set of points filtered from road or lane boundaries are fed into the geometric model fitting module. Kim [15] suggested a technique that groups those points into straight line. Huang *et al.* [12] presented the similar approach with directional information by adding a *hessian filter*. The algorithms applied by Huang *et al.* [12], Danescu and Nedevschi [9], and Jiang *et al.* [13] assign each pixel a score which represents its distance from the closest boundary point. This makes an easy way of getting an efficient fitting score for a hypothesized curve model.

Line fitting technique is the simplest geometric models used for the road boundaries which just handle straight road [6, 27, 28, 17, 3, 38]. Huang *et al.* [12], McCall and Trivedi [23] proposed the technique considering parabolic curves in birds eye-view where several assumptions can be made like the car is parallel to the road etc. Whereas Wu *et al.* [37] and Samadzadegan *et al.* [30] suggested employing parabolic curves in projective headway view, Labayrade *et al.* [19] recommended applying hyperbolic polynomial curves.

Semi-parametric models are also examined as fitted models for extracting road boundary. The advantage of those models is that they do not depend on a specific global geometry of the path. However, they easily fail with over-fitting and unrealistic path curvature.

Splines are very popular polynomial functions, widely used in representing curves. Different spline models with different properties are used to model the lane boundaries, *e.g.*, *cubic splines* [15], *b-splines* [34], *active contours (snakes)* [31], and *cubic hermite spline* [12]. The common characteristics in all spline models is that the curve is parametrized by a set of control points.

The advantage of splines over parametric models (*e.g.* parabolas) is that a small change in the parameters creates a small change in the curve; current frame model can be initialized by tracking control points from the previous frame. However, splines have cons as well. Usually, in lane detection a large number of control point candidates can be created because of many detections of lane markings. Correct selection among them is a subtle problem. Furthermore, the curve complexity increases with the increase of number of control points. Inconsistent distribution of those points also makes high curvature.

## 2.2 Road detection using temporal activities

Tracking the motion of the moving objects (vehicles) is another feature to recognize and classify road and off-road region. This approach works well in free-flowing traffic, however the difficulties arise with congestion, shadows, and lighting transitions. Furthermore, vehicles sometimes partially occlude one another and appear differently under various lighting conditions.

The technique relying on a *kalman filter* to perform vehicle tracking has been applied by Melo *et al.* [24]. Then, variable low-degree polynomials are used to model those resulting motion. Finally, *k-means clustering* technique on the coefficient space is used to obtain approximate lane centers. The algorithm presented by Lou *et al.* [20] employs the trajectories recorded in the visual tracking process that are analyzed using dynamic clustering. Coifman *et al.* [7] presented the idea of using feature-based tracking system for detecting vehicles. Instead of tracking entire vehicles, vehicle features are tracked. Features are grouped into discrete vehicles using a common motion constraint after exiting from the tracking region.

The technique proposed by Helala *et al.* [11] is built upon the concept of considering a sequence of video frames over time and identifying stable edges on the road. This accumulation of traffic frames over time augments the reliability and robustness of the algorithm in a range of traffic conditions. This work combines features, model, and activity cues.

Initially, the traffic scene is segmented into a set of regions using *superpixel segmentation*. Next, adaptive sampling with polygons approximates region contours into edges and, *hierarchical clustering* group these edges into clusters based on co-linearity. Then, confidence level is assigned to each cluster using statistical approaches called *Student-T*, *Chi-squared* distribution to rank them. After that, pairwise ranking is performed among top ranked pairs with the help of image perspective and activity. Finally, top-ranked cluster pair is declared as the dominant road boundary. They used realistic data set gathered from traffic surveillance cameras installed on the Ontario 401 highway. Their experiment includes the data set with a large variation in lighting and environmental conditions. However, the major limitation of that system is it just detects straight road boundary and is unable to handle boundary of the curved road.

The work we performed here is the combination of feature driven and model driven. It considers all the challenges mentioned in [11] and is able to handle curved roads. Our work uses the same data set as [11] for straight roads. In addition, we include traffic images with curved road to justify our algorithm. Another advantage is that our work depends on only a single traffic image instead of sequence of images. Tables 2.1, 2.2 and 2.3 compare the method proposed in this thesis against existing feature based, model based and activity based schemes, respectively.

# Chapter 3

# Methodology

We will first provide an overview of the entire procedure of the curved road boundary detection algorithm in Section 3.1. Subsequent sections will present the algorithm in detail. The identification of edge pixels using brightness, color, and texture is discussed in Section 3.2. Our data set contains low quality traffic images. Therefore, a majority of the generated edge pixels are noise. How the low quality images produce spurious edge pixels and the technique to reduce these spurious pixels are demonstrated in Section 3.3. This section also shows the way of connecting neighbouring edge pixels to build the intermediate road candidates. Section 3.4 presents how intermediate candidates are grouped together to make full length candidates. The procedure of sorting these candidates and selecting final pair as the dominant road boundary is described in Section 3.5. Finally, Section 3.6 examines the variation in the design of our road identification algorithm.

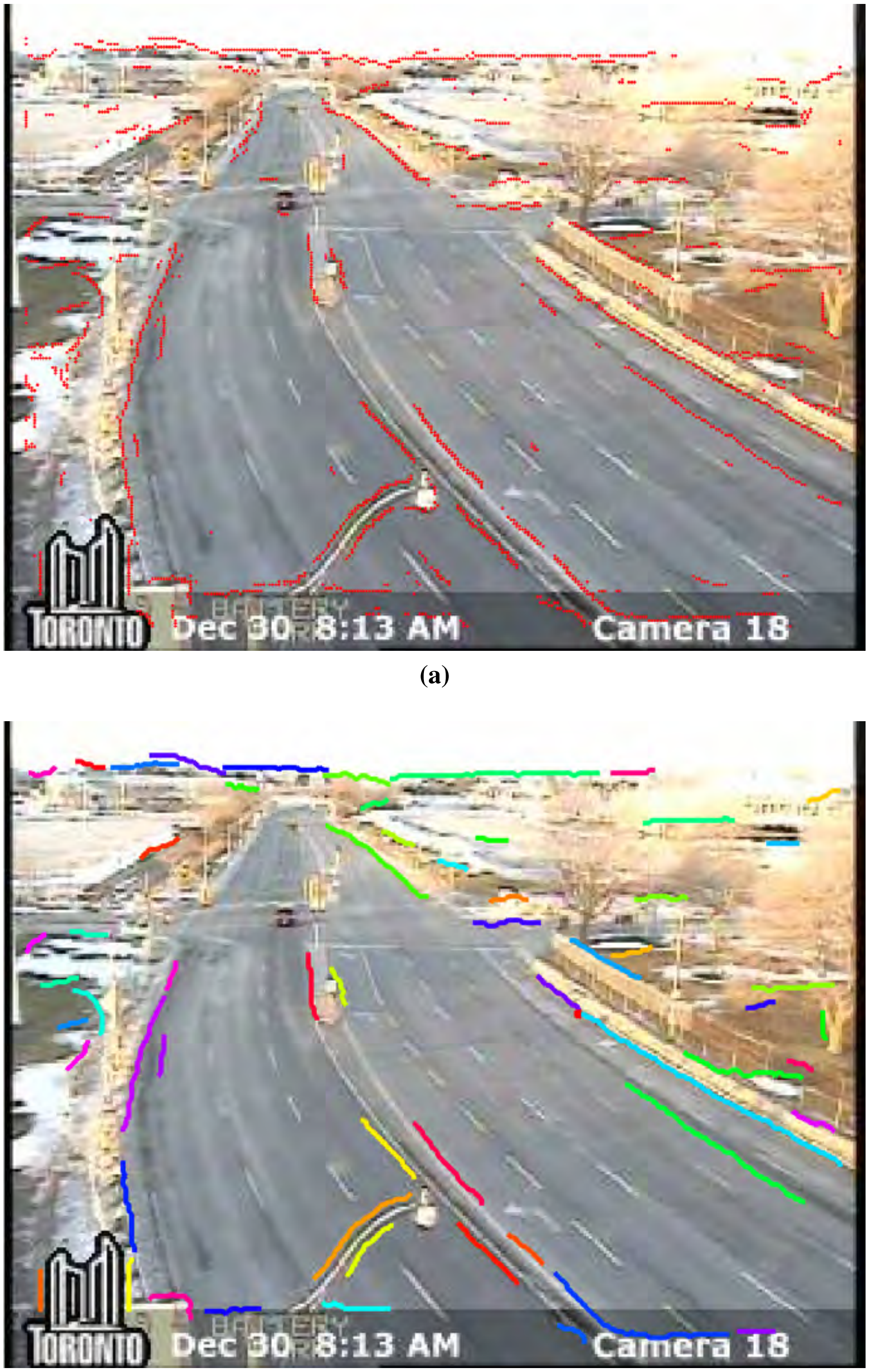
## 3.1 Overview

This section presents all the steps of the proposed technique with a concrete example, shown in Figure 3.1 to 3.3. Our proposed approach for curved road boundary detection is based on the road features and geometric structure of the road boundary. Given a traffic image, first an edge detection operation is performed to get pixel scale evidences of road boundary. For this purpose, the approach proposed by Martin *et al.* [22] is chosen that considers brightness, color, and texture features. Each of the edge pixels shown in red dot in Figure 3.1 (a), which is our evidence at pixel scale, is considered as initial candidates of the road boundary. Then, an edge linking approach proposed by Kovesi [18], described in Section 3.3 is used to connect neighboring pixel scale evidences to make road boundary evidence at intermediate scale. It also eliminates spurious edge features generated in edge detection step due to noise. In Figure 3.1 (b), different colored solid segments represent different edge segments and these are the evidences at intermediate scale. Some spurious pixel scale evidences are also removed in this step as shown in Figure (b), because of the discontinuity between edge pixels.

Next step is to perform clustering to form evidences at higher scales from the intermediate scale evidences. We chose *hierarchical clustering* which generates stronger boundary candidates by merging intermediate evidences. A few higher scale evidences are shown in Figure 3.2 (a) with light shaded segments. Members (intermediate scale evidences) of each higher scale evidence are shown by same colored solid segments. Then, a ranking algorithm is employed to rank those candidates based on their quality. Figure 3.2 (b) shows 5 top ranked candidates.

Finally, top ranked candidates are paired with each other and ranked to get the road boundaries. Usually, image features (*e.g.* color, texture *etc.*) for road and off-road regions are different. Therefore, the feature difference between the region covered by actual road boundaries and the outside region will be high. We used color feature difference between the area bounded by each pair and the outside area to rank the pair. The larger the difference, the higher the rank of the pair to be road boundaries.

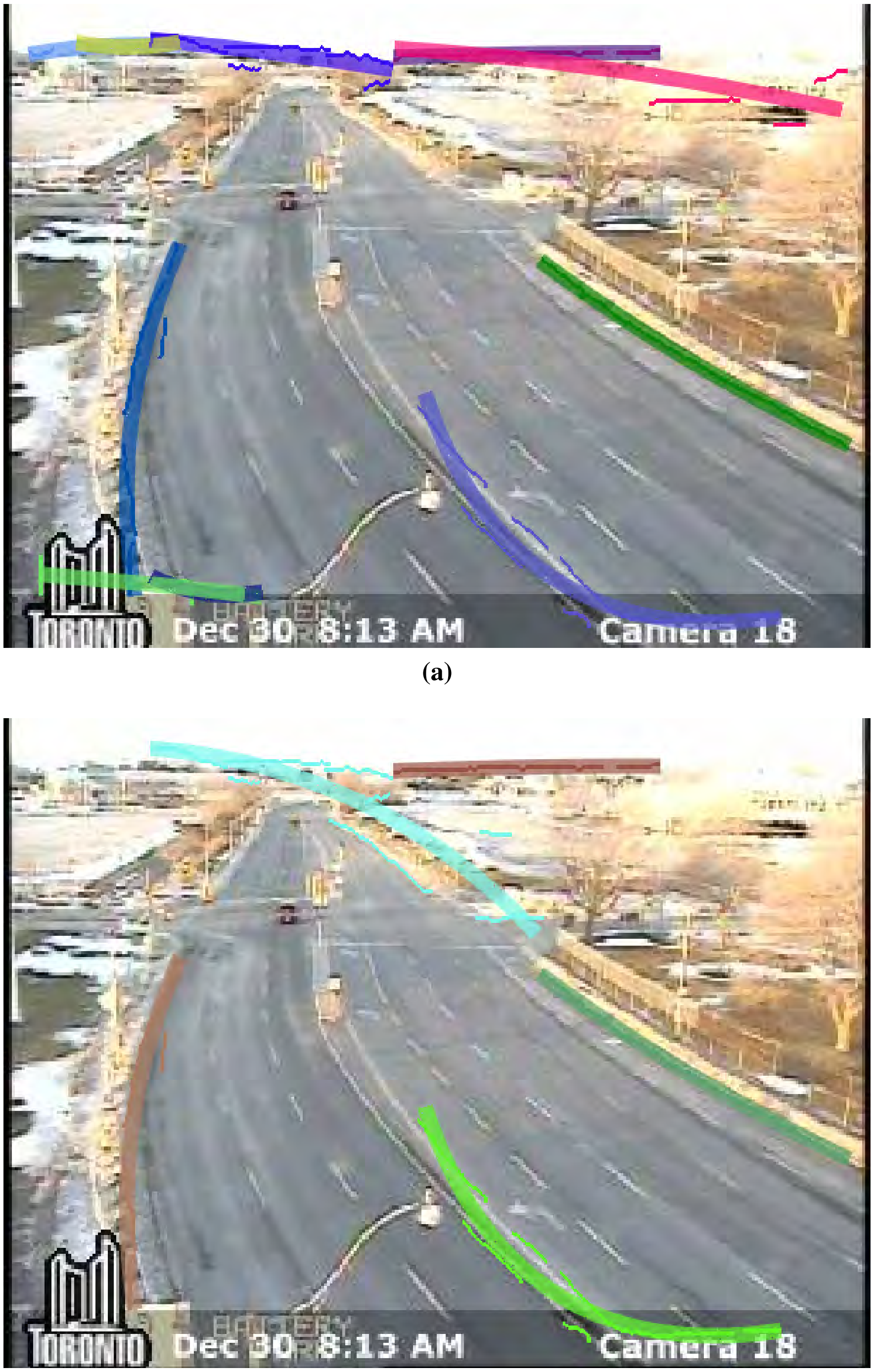
Figure 3.3 (a) and (b) show the top pair selection using this method. Here, pink and light green bars represent the color histograms for area covered by two curves and the outside area respectively. The more the overlap between two histograms, the less the feature difference between these two areas that implies lower probability to be road boundary. Green curved pair in Figure 3.3 (a) is the top pair.



**(a)**

**(b)**

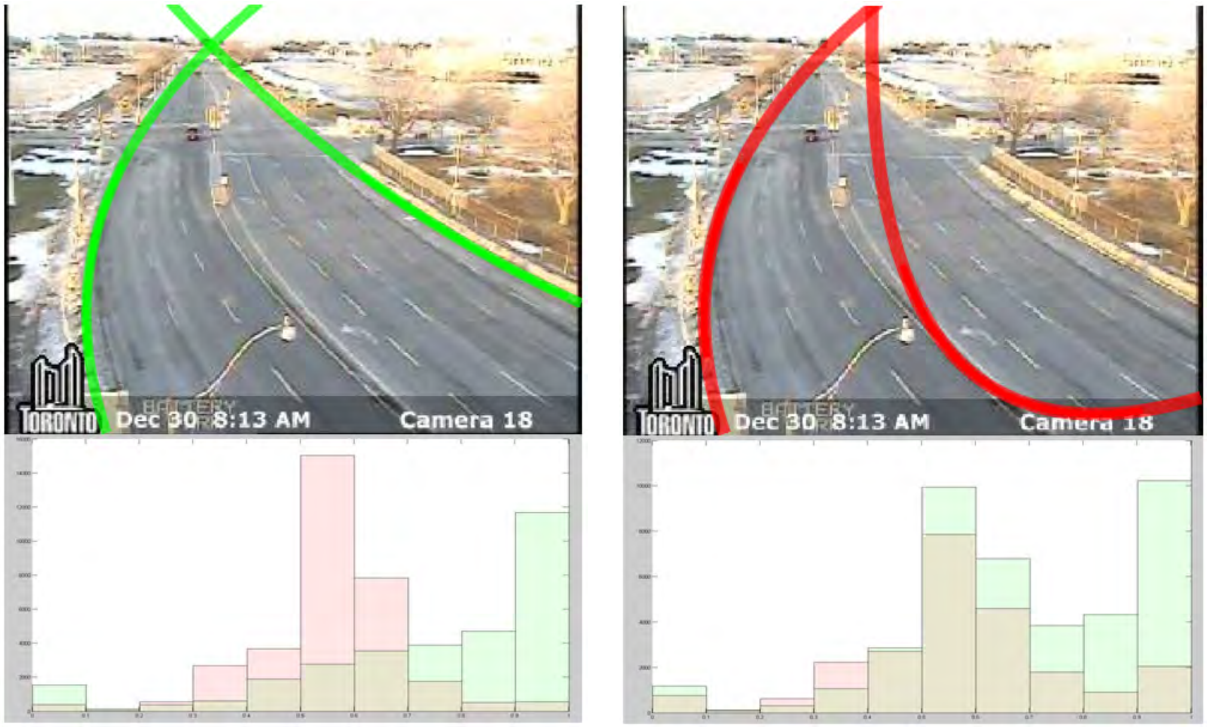
**Figure 3.1** Generating evidences at pixel and intermediate scales through (a) *edge detection* and (b) connecting neighbouring edge pixels called *edge linking* respectively.



**(a)**

**(b)**

**Figure 3.2** Generating evidences at high scale and ranking technique of the proposed approach, (a) a few of clusters generated in *hierarchical clustering*, (b) top 5 clusters.



(a) (b)

**Figure 3.3** Top pair selection, (a) and (b) histograms of the feature (color) to identify the top pair. Pink and light green bars represent the color histograms for area covered by two curves and the outside area respectively.

Each of the steps will be formally described in the following sections.

## 3.2 Generating evidences at pixel scale using edge detection

Edge detection has been used for information extraction in many computer vision applications, including object detection, object recognition, visual positioning, and tracking of object, video surveillance and so on. An edge normally referred as a sudden change in some low-level image features like brightness, color, or texture which usually illustrates boundaries of objects in a scene. Edge detection is the process of finding and localizing these sharp discontinuities in an image.

In the traffic image, road boundary divides road region from non-road region. Therefore, the low-level image features (e.g. brightness, color, texture etc.) of the boundary pixel would be different than its neighboring pixels. Those abrupt changes of pixels can be identified by any edge detection method (Rubner and Tomasi [29], Will *et al.* [36], Martin *et al.* [22] etc.) and that would be the evidences at the pixel scale for road boundary in our application. We applied the edge detection method proposed by Martin *et al.* [22] as it uses several image features like brightness, color, and texture to detect edge. However, any edge detection approach can be used to get evidences at pixel scale in this step.

Let, $I$ be an image and $I(x, y)$ be its pixel at location $(x, y)$. Let us also define a binary image $I_e$ which stores if a pixel is an edge pixel as returned by an edge detection routine. Specifically, $I(x, y)$ is an edge pixel if $I_e(x, y) = 1$. Let, $E^0 = \{(x, y)|I_e(x, y) = 1\}$ be the locations of set of edge pixels. Figure 3.4 (b) shows the output of the edge detection algorithm proposed by Martin *et al.*

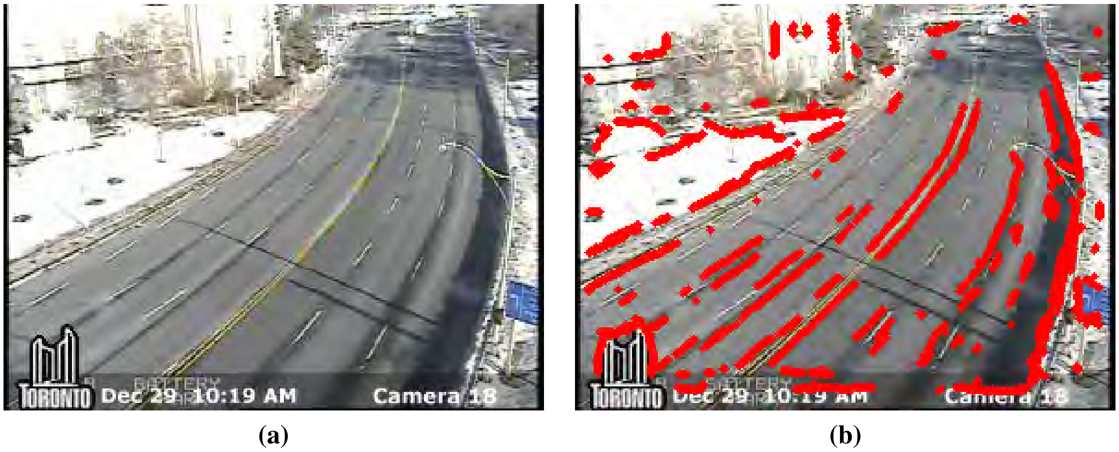
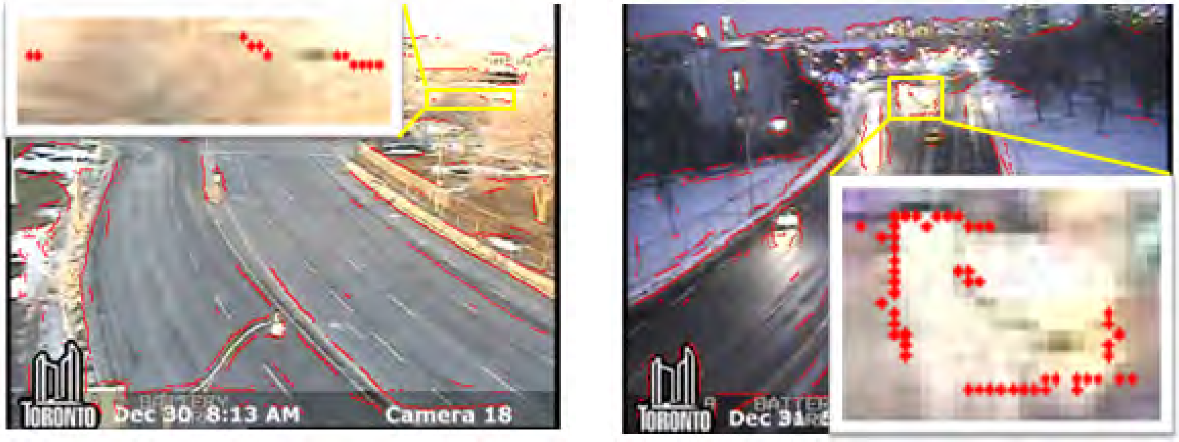

TORONTO Dec 29 10:19 AM Camera 18
TORONTO Dec 29 10:19 AM Camera 18
(a)
(b)

**Figure 3.4** Pixel scale evidence generation (a) original image, (b) same image in (a) where edge pixels are shown as red dots.

## 3.3 Preprocessing of pixel scale evidences using edge linking

### 3.3.1 Dealing with noises in images

Our data set consists of low quality traffic images. Therefore, a majority of the pixel scale evidences generated by edge detection are actually noise instead of actual boundary pixels.



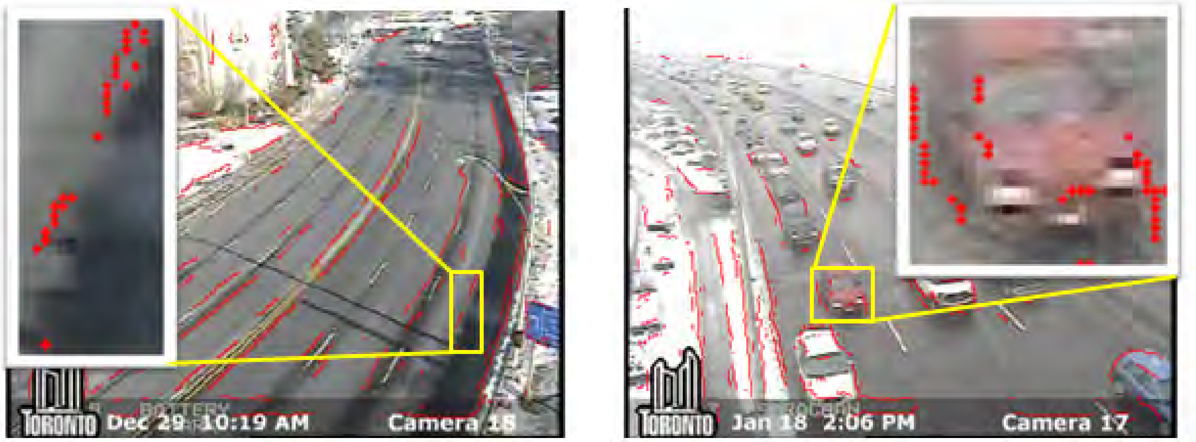
(a) (b)

**Figure 3.5** Spurious edges generated in edge detection step for (a) blurriness of the traffic image, (b) vehicle headlight reflections.

Furthermore, challenging lighting and environmental variations create more spurious edges. Figure 3.5 shows examples of those spurious edges created by blurriness (a), vehicle headlight reflections (b). Here, evidence at pixel scales are shown by red dots. Box bordered in white color is the zoomed view of the area bordered in yellow color.

Spurious edges are also generated for shadows (a) and objects on the road (b) are shown in Figure 3.6. Figure 3.7 shows spurious edges created by environmental challenge like snow (a) and challenging lighting conditions at night (b).

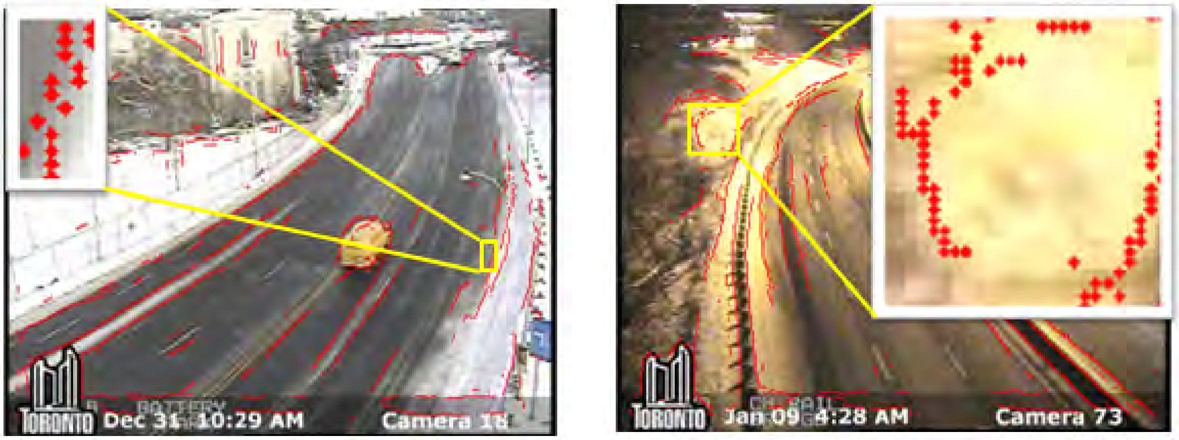
This is why we need a preliminary cleaning of those extra noisy edges. Moreover, individual edge pixel identified in edge detection step shown in Figure 3.4 (b) using red dot, is not able to express any meaningful characteristics about the image such as boundaries of the object or specific shapes like curve, line *etc*. Therefore, we need to make edge segments, where each of the segments consists of a set of edge pixels. We apply the morphological closing operation on the edges which is usually used in the process of segmentation. We call this *edge linking* which helps to prevent noisy edges or low quality evidences at pixel scale to become a candidate for road boundary. We use an open source *edge linking* technique introduced by Kovesi [18] to group neighbouring edge pixels into edge segments. Formally, an edge segment $s$ is a set of edge pixels, thus $s \in \mathcal{P}\left(E^0\right)$.



(a)

(b)

**Figure 3.6** Spurious edges generated in edge detection step for (a) shadows, (b) objects on the road.



(a)

(b)

**Figure 3.7** Spurious edges generated in edge detection step for (a) snow and (b) challenging lighting conditions at night.

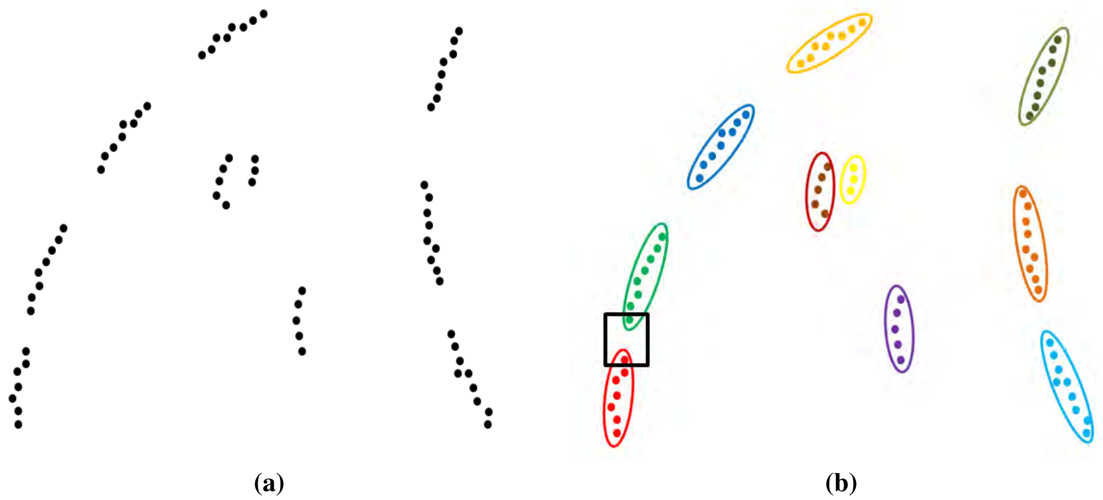
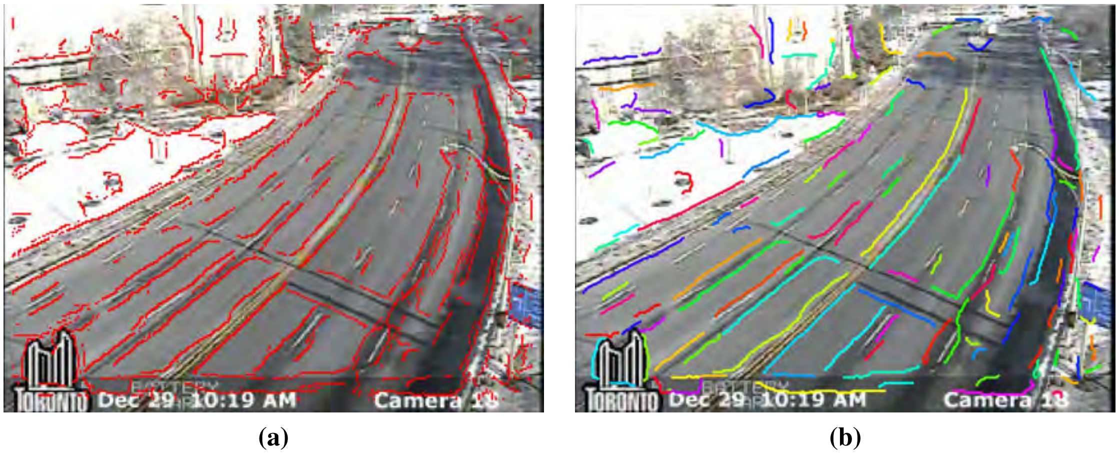


**Figure 3.8** Grouping neighbouring edge pixels using *edge linking*. (a) original image with edge pixels, (b) same image in (a) but showing edge segments in different colors.

Let, $E^1$ be the set of edge segments generated by *edge linking* approach [18]. We can write, $E^1 \subseteq \mathcal{P}\left(E^0\right)$.

In this technique, no two edge segments share a common edge pixel which implies, each edge pixel belongs to a single edge segment. Figure 3.8 (a) shows the edge pixels with black dots. In Figure 3.8 (b), edge segments are shown with different colors. Here, same colored edge pixels belong to same edge segment. We see, each edge pixel is assigned to a unique edge segment. It also considers continuity among neighbouring edge pixels. As an example, black rectangle (Figure 3.8 (b)) shows the neighbourhood between the last and first edge pixels of green and red colored edge segments. No edge pixels exist in this neighbourhood to connect these two end edge pixels. Because of this linking discontinuity two separates segments are created here.

Figure 3.9 (a) shows the edge pixels using red colors. Output of the edge linking algorithm for this image is showed in Figure 3.9 (b). Here, same colored pixels belong to same segment.



(a)

(b)

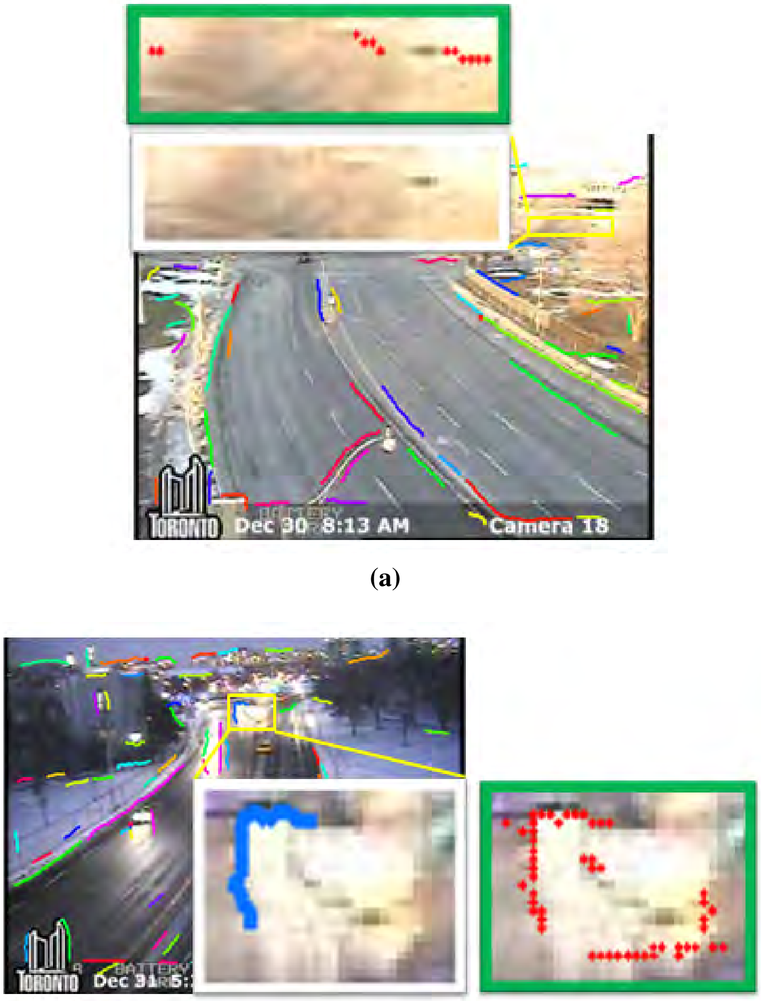
**Figure 3.9** Grouping neighbouring pixels using edge linking. (a) original image with edge pixels (b) same image in (a) but showing edge segments in different colors that are more than 10 pixels in length.

### 3.3.2 Noise reduction with edge linking

Various examples of unwanted evidences at pixel scale generated in edge detection step are shown in Figure 3.5 to Figure 3.7. Some of these noisy pixels are removed in *edge linking* method because of the linking discontinuity of edge pixels among their neighborhood. The following figures will demonstrate this.

Removing spurious edge pixels that are generated because of blurriness and vehicle headlight reflections are shown in Figure 3.10. Figure 3.11 shows examples of removing spurious edge pixels generated due to shadow and object on the road using *edge linking* approach. Figure 3.12 shows examples of removing spurious edge pixels generated due to weather difficulties like snow and challenging lighting conditions at night using *edge linking* approach.

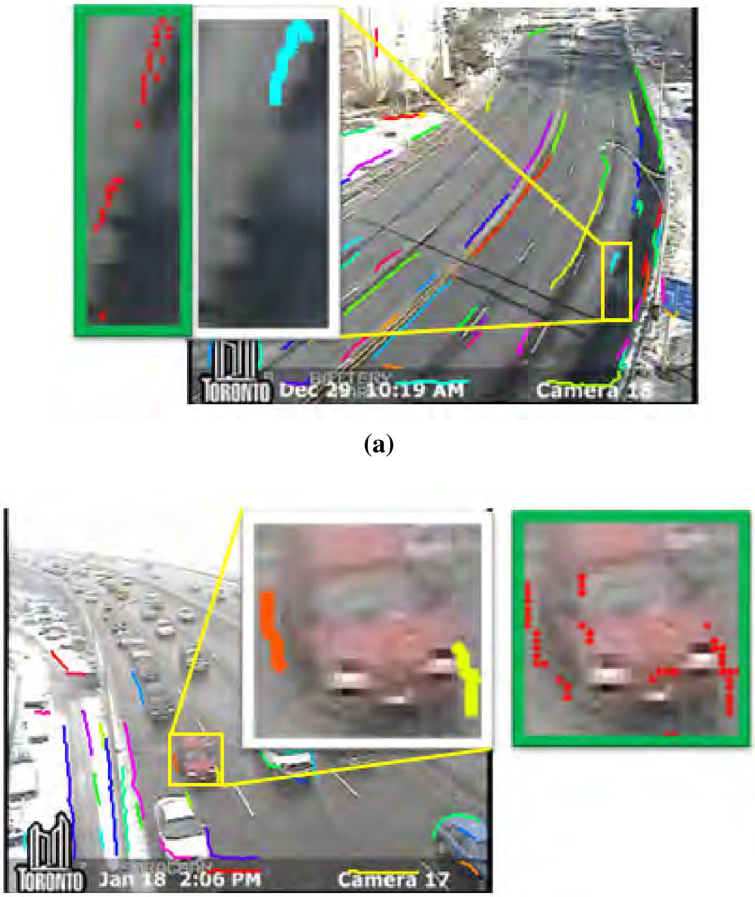
In all three figures (Figure 3.10 to Figure 3.12), the box bordered in white color is the zoomed view of the area bordered in yellow color in the image. Edge pixels of corresponding area are shown in red colored dot within the box bordered in green color. Each colored segment within white box represents evidence at intermediate scale which links several pixel scale evidences of that area.



**(a)**

**(b)**

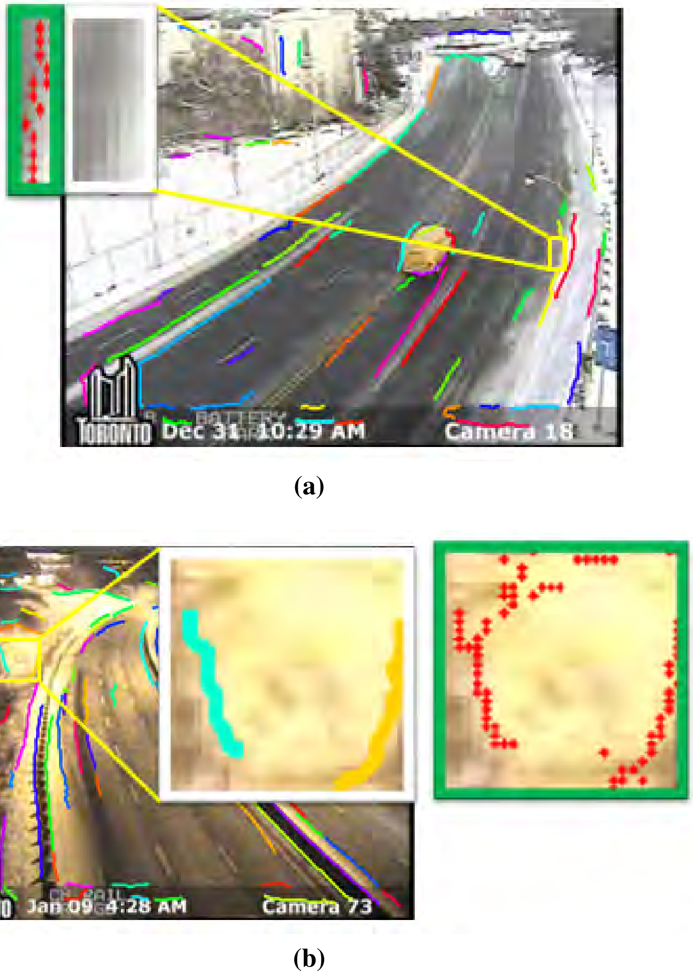
**Figure 3.10** Reduction of spurious edge pixels shown in Figure 3.5 using *edge linking* for (a) blurriness of the traffic image, (b) vehicle headlight reflections. Here, the box bordered in white color is the zoomed view of the area bordered in yellow color in the traffic image. Each colored segment within the white box represents evidence at intermediate scale which links several pixel scale evidences of that area. Pixel scale evidences of corresponding area are shown in red colored dot within the box bordered in green color.



(a)

(b)

**Figure 3.11** Reduction of spurious edge pixels shown in Figure 3.6 using *edge linking* for (a) shadows, (b) objects on the road. Here, the box bordered in white color is the zoomed view of the area bordered in yellow color in the traffic image. Each colored segment within the white box represents evidence at intermediate scale which links several pixel scale evidences of that area. Pixel scale evidences of corresponding area are shown in red colored dot within the box bordered in green color.



(a)

(b)

**Figure 3.12** Reduction of spurious edge pixels shown in Figure 3.6 using *edge linking* for (a) snow, (b) challenging lighting conditions at night. Here, the box bordered in white color is the zoomed view of the area bordered in yellow color in the traffic image. Each colored segment within the white box represents evidence at intermediate scale which links several pixel scale evidences of that area. Pixel scale evidences of corresponding area are shown in red colored dot within the box bordered in green color.

## 3.4 Generating evidences at high scale

The goal of this section is to produce high scale evidences from evidences at intermediate scale generated in previous section.

### 3.4.1 Hierarchical clustering

Clustering is a popular approach for statistical data analysis in machine learning, pattern recognition, image analysis, and many other fields. The main purpose is to group a set of objects into subsets called 'clusters' in such a way that the objects within each cluster are more similar to one another than objects allocated to different clusters. A popular clustering approach is *hierarchical clustering* which builds a hierarchy tree of clusters. Given a set of $n$ items $\{s_i|i = 1...n\}$ to be clustered, it starts by creating a cluster $c_i$ for each item $s_i$. All of these initial clusters $c_i$ are formed the leaf level of the hierarchy tree. It then merges the closest pair of clusters to form a single cluster and move up the hierarchy with one less number of clusters. This process continues until it reaches to a single cluster called root of the hierarchy tree.

Set of edge segments $E^1$ generated in *edge linking* step are fed into the *hierarchical clustering* as leaf level of the hierarchy tree. Let, $C^k$ is the set of clusters at level $k$ in the hierarchy tree. Therefore, $C^0 = E^1$. Let, the cluster $c \in C^k$, where we can write, $c \in \mathcal{P}\left(E^0\right)$. At each level $k$ two clusters $c, c' \in C^k$ with minimum in-between distance are merged into a single cluster and move up the hierarchy with one less cluster. Therefore, $C^k \subseteq \mathcal{P}\left(E^0\right)$. Assume, $\mathbb{C}$ is the set of all clusters generated in *hierarchical clustering* and can be defined as

$$\mathbb{C} = \bigcup_{k=0}^{n} C^k \tag{3.1}$$

Here $n = |E^1| - 1$.

Figure 3.13 shows how *hierarchical clustering* builds based on the output from *edge linking*. Generally original road boundaries belong to two clusters of the hierarchy tree.

We define the *hierarchical clustering* algorithm inductively.

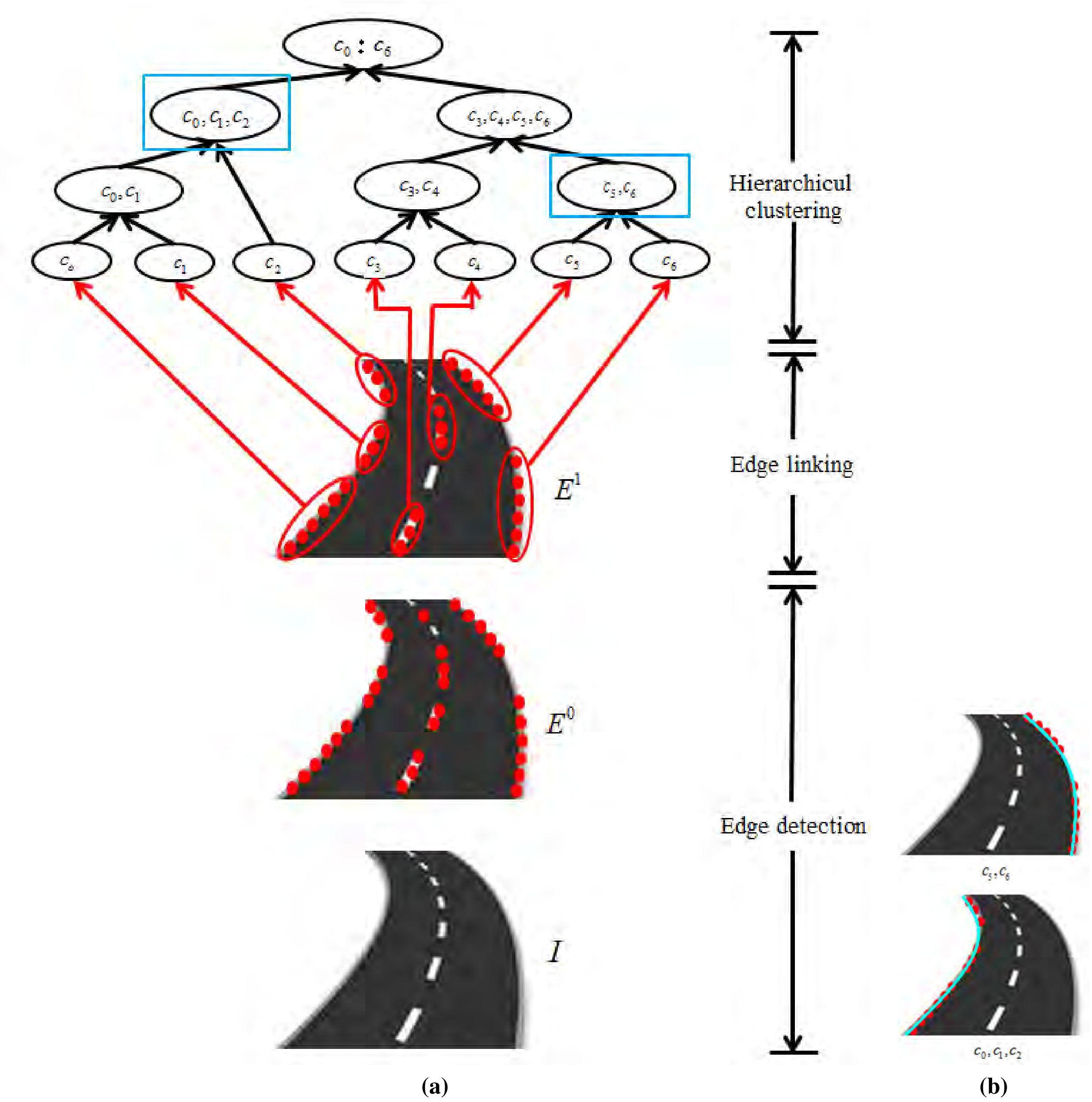


**Figure 3.13** (a) Hierarchical clustering from given traffic image, (b) clusters belong to original road boundaries shown in (a) with cyan color rectangles .

- ***Base Case***:

  $C^0 = E^1$

- ***Induction Step(s)***:

  Suppose we have formed $C^k$ and $|C^k| > 1$. Let, $d$ be a dissimilarity measure of $C^k$. Let, $c, c' \in C^k$ be two distinct clusters such that $d(c, c')$ is minimal, then

  $$C^{k+1} = C^k - \{c, c'\} \cup \{c \cup c'\}$$

  The induction terminates when $|C^k| = 1$.

In next section, we will describe how to calculate dissimilarity between tow clusters $d(c, c')$.

### 3.4.2 Dissimilarity measure

Consider two clusters $c, c' \in \mathbb{C}$, we wish to define a dissimilarity measure $d(c, c') \geq 0$ such that

- If $c$ and $c'$ should be part of same road boundary, then our objective is to have $d(c, c')$ small and
- If $c$ and $c'$ are parts of different road boundaries, then $d(c, c')$ should be large.

To achieve this, we hypothesize that $c$ and $c'$ are parts of the same road boundary. Thus, there should be a curve that fits $c \cup c'$ well. Our goal is to design the dissimilarity measure $d(c, c')$ to reflect how well a continuous curve can fit the pixels in $c \cup c'$ . We do so by considering four aspects of curve fitting:

- Fit error;
- Difference in *arc length* and *Euclidean length*;
- Accumulative curvature; and

- Fitted curve discontinuity.

Let, $X = c \cup c'$. Assume, $f$ be a function which maps $X$ to a $2D$ parametric curve. Now, $f(X) = \vec{h}$ where $\vec{h} : [0,1] \to \mathbb{R}^2$ and $\vec{h}$ is continuous and differentiable, where $\vec{h}(t) = [h_x(t), h_y(t)]$.

Note: $d(c, c')$ will be evaluated based on $\vec{h}$ and $X$.

**Fit error** $(d_{\text{fit}-\text{error}})$

In our proposed approach, the most important dissimilarity calculation feature is polynomial fit error. A $2^{nd}$ degree polynomial is fitted to the merged clusters of $c$, $c'$ and checked the fit error to express dissimilarity between those two clusters.

Let $\vec{p} \in X$, define $\text{proj}(\vec{p}, \vec{h})$ as the projected point of $\vec{p}$ on to $\vec{h}$; given by:

$$\vec{h}(\underset{t \in [0,1]}{\arg\min} \|\vec{p} - \vec{h}(t)\|)$$

Therefore, we can define fit error as

$$d_{\text{fit}-\text{error}} = \sqrt{\sum_{\vec{p} \in X} \|\vec{p} - \text{proj}(\vec{p}, \vec{h})\|^2} \tag{3.2}$$

In Figure 3.14, pixels of a cluster are shown by red dots. Blue circles are the pixels belong to the fitted curve. Here, total polynomial fit error will be $\sum l_i$.

Greater error indicates higher dissimilarity between clusters which are merged as candidate cluster for next level of the hierarchy. In Figure 3.15, members of two different clusters are shown by red and purple color dots and fitted curve is shown by solid green color. Figure 3.15 (a) shows a case where fitted curve has enormous amount of error as two clusters belong to just opposite side of the road. In contrast, Figure 3.15 (b) shows a case where fitted curve has very tiny polynomial fit error and a good candidate to merge to make next level of the hierarchy.

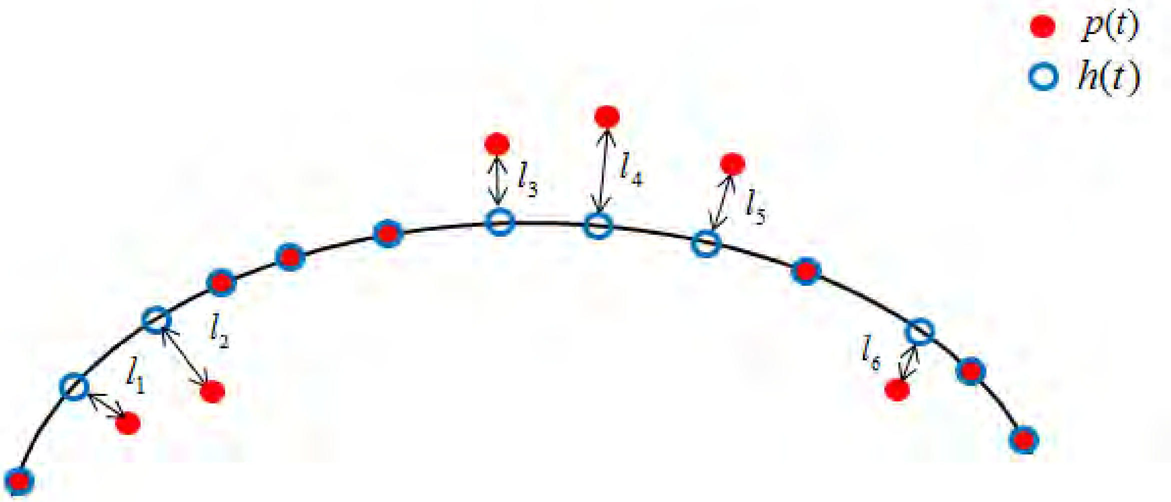


**Figure 3.14** Polynomial fit error calculation.

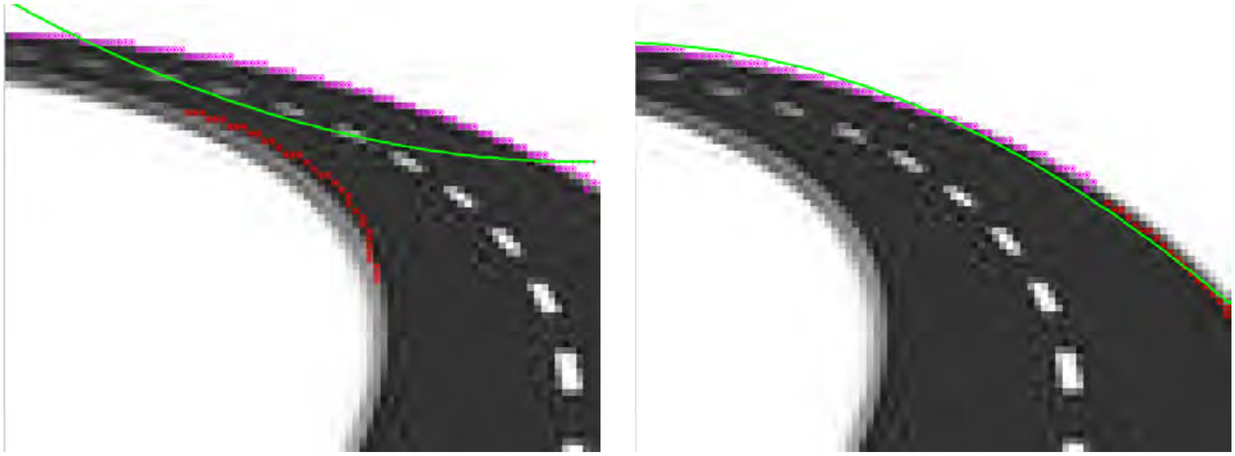


**Figure 3.15** (a) bad fit of the 2nd degree polynomial and (b) good fit of the 2nd degree polynomial to the pixels from two clusters.

### Difference in arc and Euclidean length $(d_{\text{diff}-\text{length}})$

Let, $L_0$ be the *Euclidean length* of $\vec{h}$ and can be defined as

$$L_0 = \|\vec{h}(1) - \vec{h}(0)\| \tag{3.3}$$

Let, $L_1$ be the *arc length* of $\vec{h}$ and can be expressed as

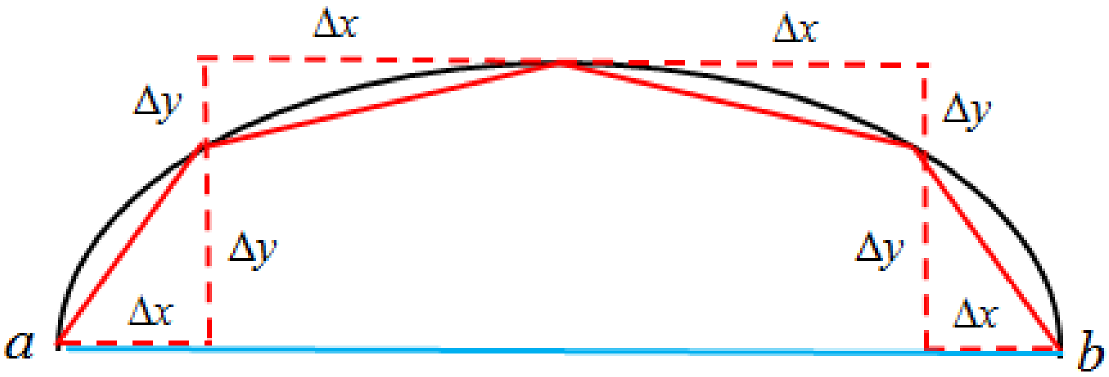


**Figure 3.16** *Arc* and *Euclidean length* difference.

$$L_1 = \int_0^1 \sqrt{\left(\frac{dh_x}{dt}\right)^2 + \left(\frac{dh_y}{dt}\right)^2} dt \tag{3.4}$$

Therefore, the *arc* and *Euclidean length* variation is

$$d_{\text{diff}-\text{length}} = |L_1 - L_0| \tag{3.5}$$

*Arc* and *Euclidean length* difference indicates the curvature of the curve. Larger difference indicates higher curviness and too much curviness indicates bad cluster as our dataset contains traffic images with medium curvy roads. Therefore, *arc* and *Euclidean length* difference helps to avoid grouping two clusters if the fitted curve of these clusters has huge curvature. This is an assumption and therefore our technique fails on road boundaries that exhibit large curvature.

Figure 3.16 shows the difference between *arc* and *Euclidean length*. Blue color line is the *Euclidean* and summation of the red color solid lines is the *arc length* between two points $a$ and $b$.

### Accumulated curvature $(d_{\text{curvature}})$

We wish to penalize curves with large curvature (as roads usually have gentle curvatures).

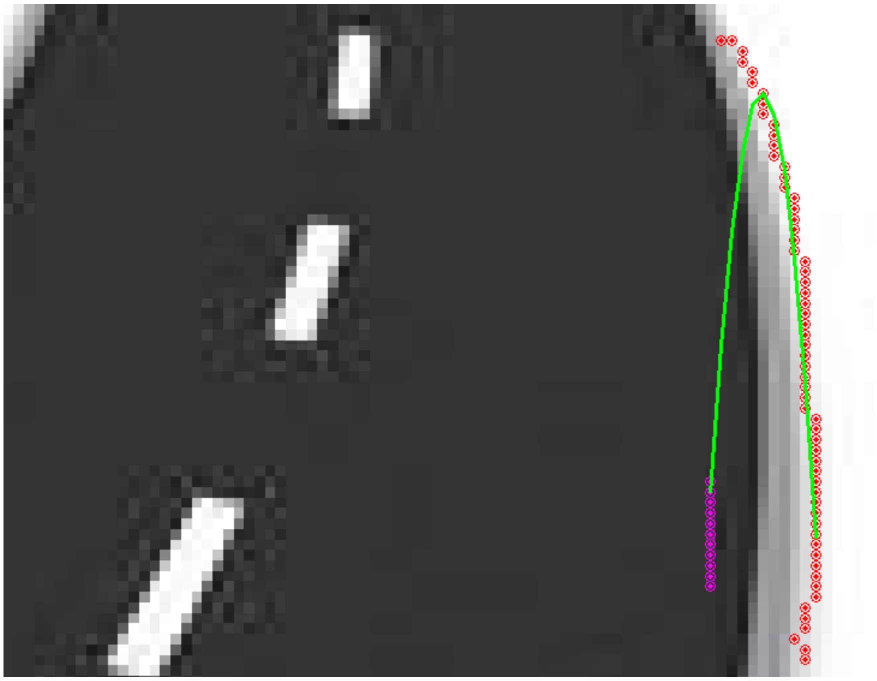


**Figure 3.17** Fitted curve curvature.

Recall the curvature of $\vec{h}$ at some $t$ is defined as $\frac{d\theta/dt}{ds/dt}$ where, $\theta$ is the angle of tangent line at $\vec{h}(t)$ and $s$ is the *arc length* from $0$ to $t$.

It is known that

$$\text{curvature}(t) = \left| \frac{h'_x h''_y - h'_y h''_x}{((h'_x)^2 + (h'_y)^2)^{3/2}} \right| (t) \tag{3.6}$$

Therefore, curvature of $\vec{h}$ will be

$$d_{\text{curvature}} = \int_0^1 \text{curvature}(t)dt \tag{3.7}$$

The shape of a curve depends very largely upon the curvature. The unit tangent vector changes very little, if a curve keeps close to the same direction and produces small curvature. On the other hand, large curvature is produced, where the curve takes a huge turn.

In Figure 3.17, two clusters are shown by red and purple color dots and fitted curve is shown by green color segment. We can say they are not good candidates to merge as purple color cluster does not belong to original road boundary. Here, tow clusters are place side by

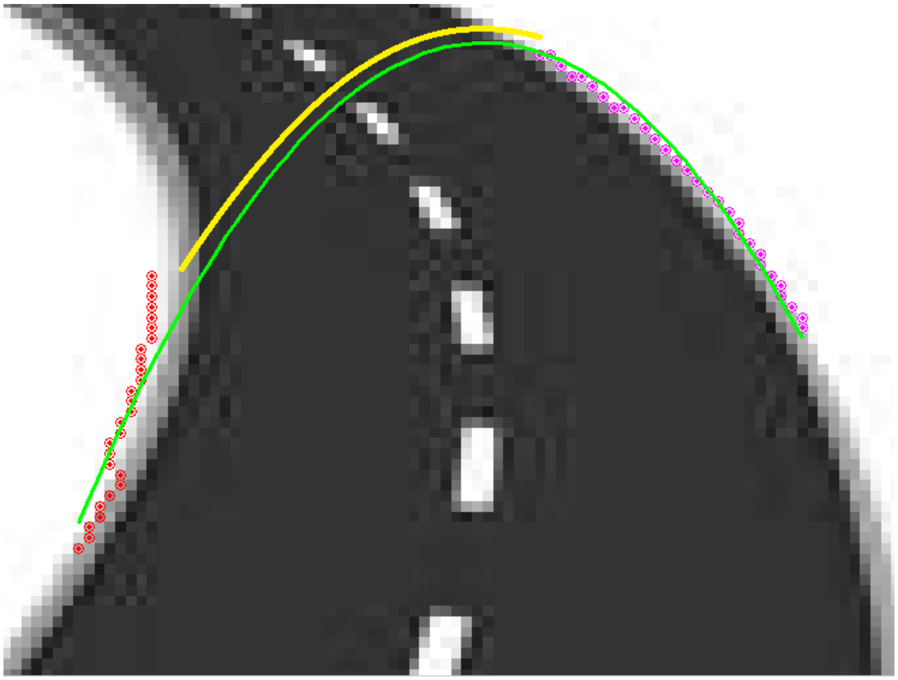


**Figure 3.18** Example of fitted curve continuity.

side. Hence, the fitted curve has large curvature and will produce larger dissimilarity. Therefore curvature helps to prevent merging these clusters for next level of the hierarchy tree. Usually, fitted curve which has immense curvature leads to wrong merging of clusters based on our assumption that roads have gentle curvature.

### Fitted curve continuity $(d_{\text{gap}})$

Only fit error and curvature sometimes lead to wrong choice of cluster as shown in Figure 3.18. Here, red and magenta color pixels represent two clusters and green color curve represents fitted curve to these two clusters.

In this case fit error will be small as most of the pixels from both clusters fitted nicely to the curve although they belong to opposite side of the road boundary. However, the fitted curve is discontinuous here. No pixels from both clusters cover the yellow portion of the curve. So, curve discontinuity will give a enormous error.

Consider the projected pixel points on $\vec{h}$ as

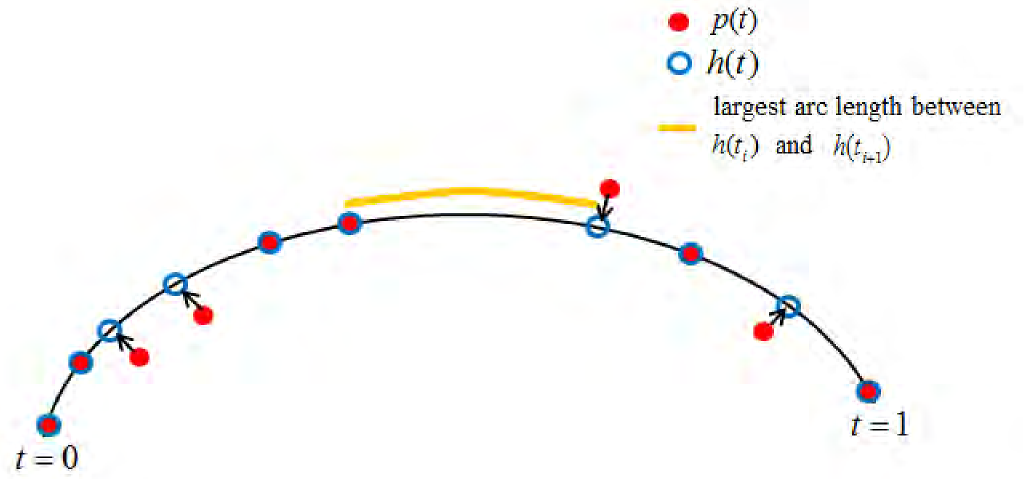


**Figure 3.19** Fitted curve discontinuity as largest *arc length* between the adjacent projected pixels. Here, blue circles are the projected points on the fitted curve for cluster points shown by red dots.

$$Y = \{\text{proj}(\vec{p}, \vec{h}) : \vec{p} \in X)\} \tag{3.8}$$

Sort $Y$ into $\{q_1, q_2, q_3, ..., q_N\}$ such that $\vec{q_i} = \vec{h}(t_i)$ with increasing $t_i$.

Therefore the discontinuity of the curve will be largest *arc length* between the adjacent projected pixels $\vec{p}$ on $\vec{h}$ and can be defined as

$$d_{\text{gap}} = \underset{i \in [1,N]}{\arg\max} \int_{t_i}^{t_{i+1}} \sqrt{\left(\frac{dh_x}{dt}\right)^2 + \left(\frac{dh_y}{dt}\right)^2} dt \tag{3.9}$$

In Figure 3.19, orange color segment represents the discontinuity of the fitted curve which is the largest *arc length* among all adjacent projected pixels of this curve. Here, red dots are the members at pixel scale from two clusters to be merged and blue circles are the corresponding projected points on the fitted curve.

Finally, total dissimilarity which is used to find closest pair among all pairs in each level of the hierarchy tree can be calculated as

$$d(c, c') = d_{\text{fit}-\text{error}} + d_{\text{diff}-\text{length}} + d_{\text{curvature}} + d_{\text{gap}} \tag{3.10}$$

Normalized values are used here for each of the four dissimilarity aspects in each level of the hierarchy tree.

## 3.5 Ranking and top pair selection

As we stated earlier, *hierarchical clustering* generates large number of clusters as candidate road boundary. However, many of those clusters could lead wrong path as boundary. Therefore, it is necessary to build a ranking function to sort those clusters by their quality which is suitable in our application. This section ranks those clusters and then pairs top $k$ ranked clusters with each other to perform pairwise ranking. Finally, it declares top ranked pair as road boundaries.

### 3.5.1 Ranking function

Let, $d_{\text{fit}-\text{error}}$ be as defined in Equation 3.2. We want to design a scoring function $R : \mathbb{C} \to \mathbb{R}^+$ that assigns a score to each cluster $c \in \mathbb{C}$ in the hierarchical clusters. The objective is that desirable clusters require higher scores. To that end we wish $R$ to :

- Clusters with small fit error;
- Clusters with large arc length; and
- Clusters with large number of evidences at pixel scale.

We find the following simple scoring function perform well:

$$R(c) = \frac{\exp^{\alpha(L_1(c)-\beta)}\,|c|}{d_{\text{fit}-\text{error}}(c)} \tag{3.11}$$

Here, $\alpha$ and $\beta$ are constants. We related,

$$\beta = \arg\min_{(c)\in\mathbb{C}} L_1(c)$$

$$\alpha = \frac{3}{\arg\max_{c \in \mathbb{C}} L_1(c) - \arg\min_{c \in \mathbb{C}} L_1(c)}$$

We also used normalized value for $d_{\text{fit}-\text{error}}$ and $|c|$.

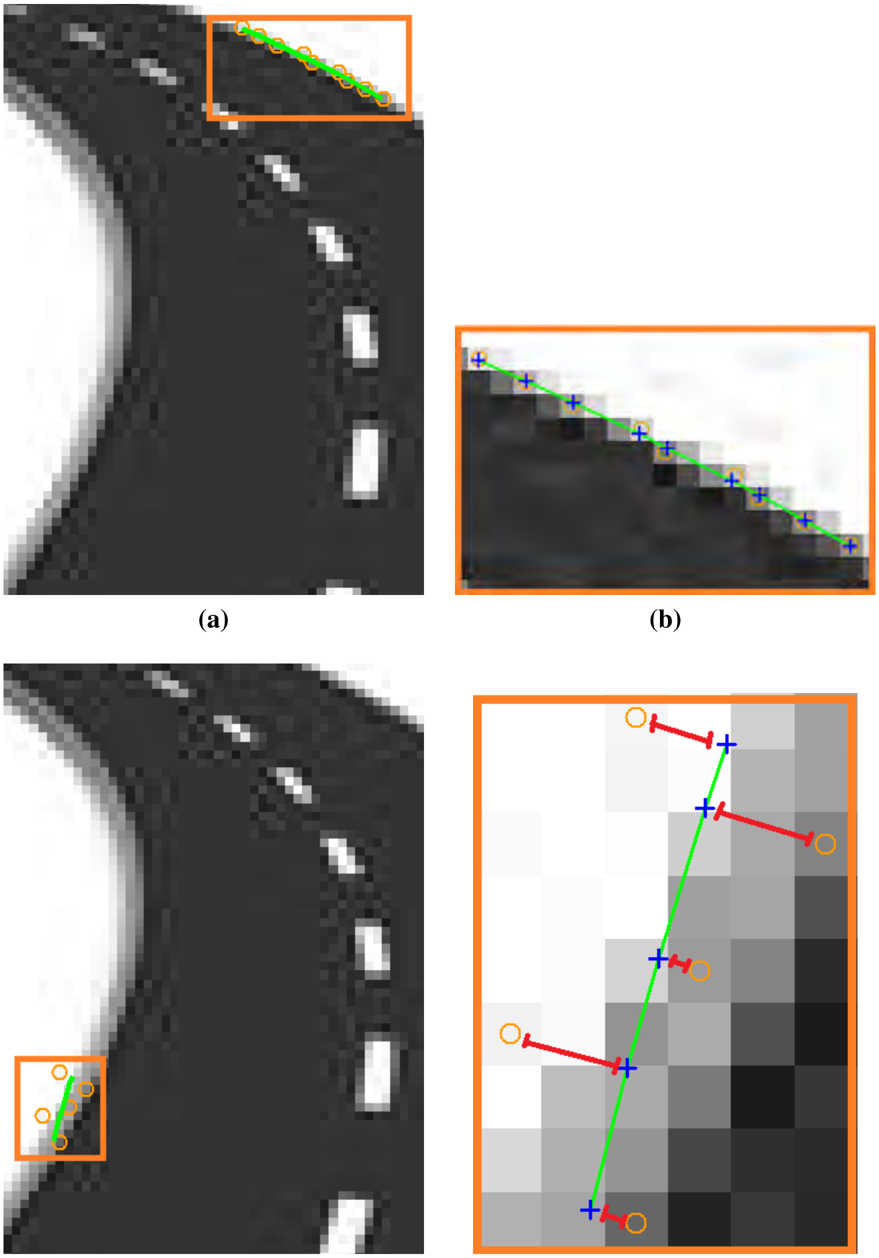
Figure 3.20 (a) shows a cluster with members at pixel scale represented by orange color circles. Green segment is the fitted curve to it. Here, blue color positive signs are the members of the fitted curve (Figure 3.20 (b)). The fit error is very tiny since original and polynomial fitted members are almost overlapped. Also, its *arc length* is longer and number of members is higher compared to the cluster shown in (c). In contrast, the polynomial fitted poorly to the cluster in (d) (zoomed version of (c)). Here, the *Euclidean distance* between members of the cluster and fitted curve is shown with red color line. The ranking function will give higher rank to the first cluster than $2^{nd}$ one which is preferable for our application. Figure 3.21 shows sorted (lower to higher) ranked clusters with lighter to darker red color curves using the proposed ranking function.

### 3.5.2 Pairwise ranking

While the scoring function $R : \mathbb{C} \to \mathbb{R}^+$ allows us to identify the top candidates for curved road boundaries, we are still left with the talk of identifying the actual region of the road. Thus, we need to assign an algorithm for selecting pairs of clusters which are the two edges of the road. Let, $\text{top}_k(\mathbb{C})$ be the top $k$ clusters ranked by $R$.

#### Filtering by vanishing point

Given two clusters $c, c'$, we wish to determine if their geometric orientation is compatible with camera perspectives of the road. See Figure 3.22, shows 4 pairs of curves with gray and black solid color. In this figure, two black pairs satisfy the perceptiveness and two gray pairs do not. Recall we fit two curves to the pixel scale evidences in $c$ and $c'$ respectively. Let, $f$ and $g$ be the parametric curves fit to $c$ and $c'$ respectively. First we approximate $f$ and $g$ by two straight lines. Assume, $u(t)$ is the straight line between $f(0)$ and $f(1)$. $v(t)$ is the straight line



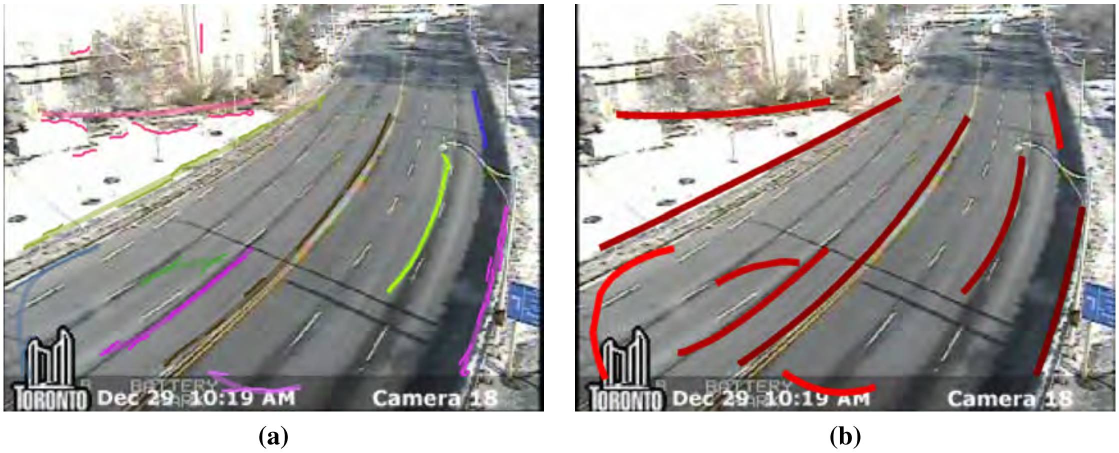
(a)

(b)

(c)

(d)

**Figure 3.20** Example of how ranking function works, (a) better ranked cluster with tiny fit error, longer length, and higher no. of members compared to (c), (b) zoomed version of (a), (c) lower ranked cluster than (a), and (d) zoomed version of (c).


TORONTO Dec 29 10:19 AM Camera 18
TORONTO Dec 29 10:19 AM Camera 18

(a)

(b)

**Figure 3.21** (a) Transparent curves are clusters with members represented by same colored solid segments, (b) ranking order (lower to higher) from lighter to darker red.

between $g(0)$ and $g(1)$. Let, $w = u(t) \cap v(t)$ be the intersection of $u$ and $v$, commonly known as vanishing point [8]. Due to camera configuration, we assert that

$$w_y \geq u(1) \text{ and } w_y \geq v(1) \tag{3.12}$$

Now, if $f$ and $g$ are such that Equation 3.12 is true, then we say $c$ and $c'$ are perspective compatible. This compatibility allows us to effectively identifying all candidate pairs for road region as

$$P = \{(c, c') : c \in \text{top}_k(\mathbb{C}), c' \in \text{top}_k(\mathbb{C}) \text{ and } c, c' \text{ are perspective compatible}\} \tag{3.13}$$

Figure 3.22 shows 2 pairs of curves with gray and black solid color. A line is drawn between two end points of each curve as shown in gray and black color dashed lines. Then, for each curved pair, the vanishing point (shown by black and gray dots) is calculated based on their corresponding lines. In this figure, the black pair is perspective compatible and the gray pair is filtered out from top ranked pairs as it does not satisfy the perspective compatibility.

Although perspective compatibility reduces number of top candidate pairs, still need to sort

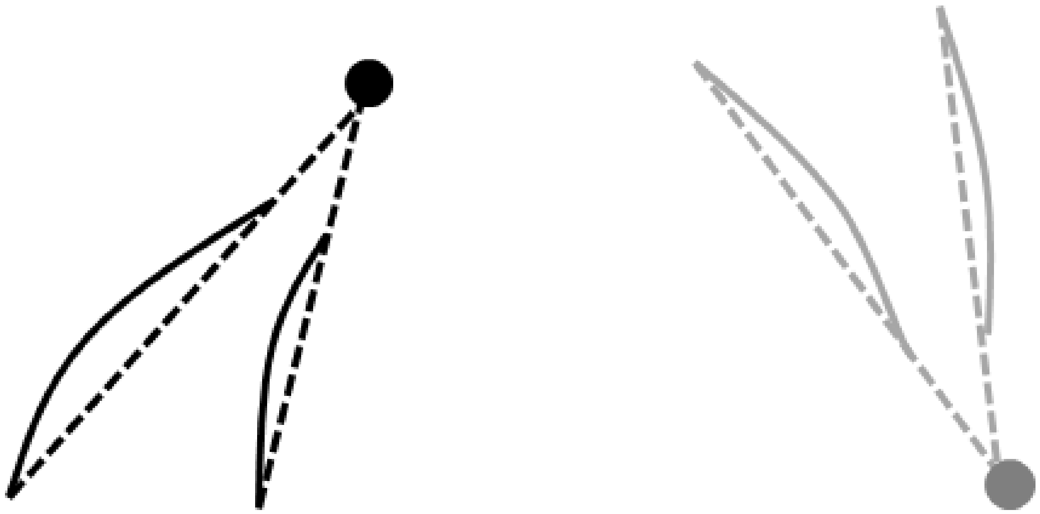


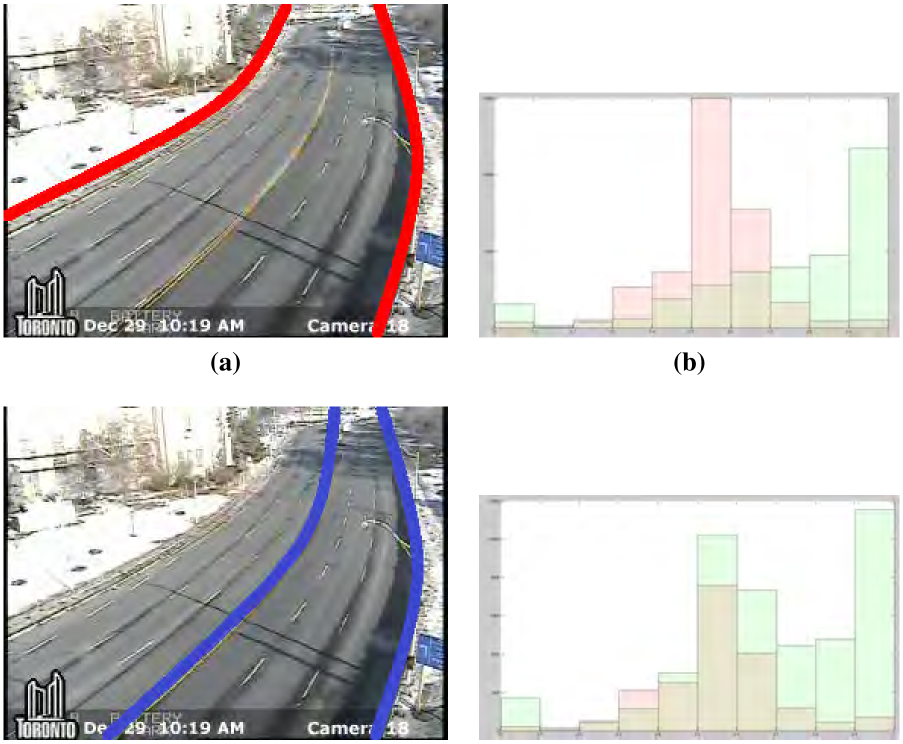
**Figure 3.22** The use of perspective cue for filtering cluster pairs. Here, solid black curves and gray curves indicate cluster pairs which is survived and rejected in perspective filtering respectively.

the remaining pairs and find the top pair as curved road boundaries which is the final aim of our application.

### Bhattacharyya distance

*Bhattacharyya distance* measures the dissimilarity of two distributions or samples. Generally, it approximates the amount of overlap between two samples or populations. More the overlap, lower the distance. We use the image color feature to measure the dissimilarity between the area of each top ranked pair and the outside area through *Bhattacharyya distance*. In our application, road boundary separates the road region from non-road region. Usually image features of the road region are different than other regions. Hence, features of the area covered by top pair clusters must be most distant from the outside area among all ranked pairs generated in previous step. Therefore, *Bhattacharyya distance* will be highest for top pair through all ranked pairs. Figure 3.23 shows an example of the potentiality of this technique.

The distance also depends on the number of partitions which implies the interval of the sample data. Choosing correct interval is necessary as too few partitions overemphasizes the



(a) (b)

(c) (d)

**Figure 3.23** The use of *Bhattacharyya* distance to get top rank pair, (a) good pair, (b) color histogram of (a). Light green and light pink bars are color histogram for the area within 2 curves and outside area respectively. Overlapped bars are shown by gray color. (c) bad pair, and (d) color histogram of (c) which has more overlapped region than (a), ensures lower *Bhattacharyya* distance and lower score than (a).

overlap region, and too much partitions sometimes do not create reasonable amount of overlapped region by putting almost each member of the sample to individual partitions. We use LAB (a three dimensional color model where the dimension $L$ for light to dark, $a$ for red to green, and $b$ for blue to yellow color) color histogram and 32 number of bins to calculate the *Bhattacharyya distance*.

Therefore, the area bounded by each $(c, c') \in P$ is ranked using *Bhattacharyya distance* as

$$B(c, c') = \sqrt{1 - \sum_{s=1}^{n} H^{in}(s) H^{out}(s)} \tag{3.14}$$

Here, $H^{in}$ and $H^{out}$ are the normalized color histogram for area bounded by $(c, c')$ and the outside area respectively. $n$ is the number of bin.

Finally, the pair with maximum *Bhattacharyya distance* is declared as dominant road boundary and can be defined as

$$(c, c') = \arg\max_{(c,c') \in P} B(c, c') \tag{3.15}$$

## 3.6 Choices of fitting algorithms

### 3.6.1 Elliptical fitting

An ellipse is a special case of a general conic which can be described by an implicit second order polynomial

$$f(\mathbf{a}, \mathbf{x}) = \mathbf{a}.\mathbf{x} = ax^2 + bxy + cy^2 + dx + ey + f = 0 \tag{3.16}$$

with an ellipse specific constraint: $b^2 - 4ac < 0$. Where, $\mathbf{a} = [a\ b\ c\ d\ e\ f]^T$ and $\mathbf{x} = [x^2\ xy\ y^2\ x\ y\ 1]$. $f(\mathbf{a}, \mathbf{x})$ is called the algebraic distance of a point $(x, y)$ to the given conic.

Generally, the fitting of a general conic to $n$ data points is produced by minimizing the sum of squared algebraic distances to those points and can be represented as

$$\mathcal{D}_A(\mathbf{a}) = \sum_{i=1}^{n} f(\mathbf{x}_i)^2 \tag{3.17}$$

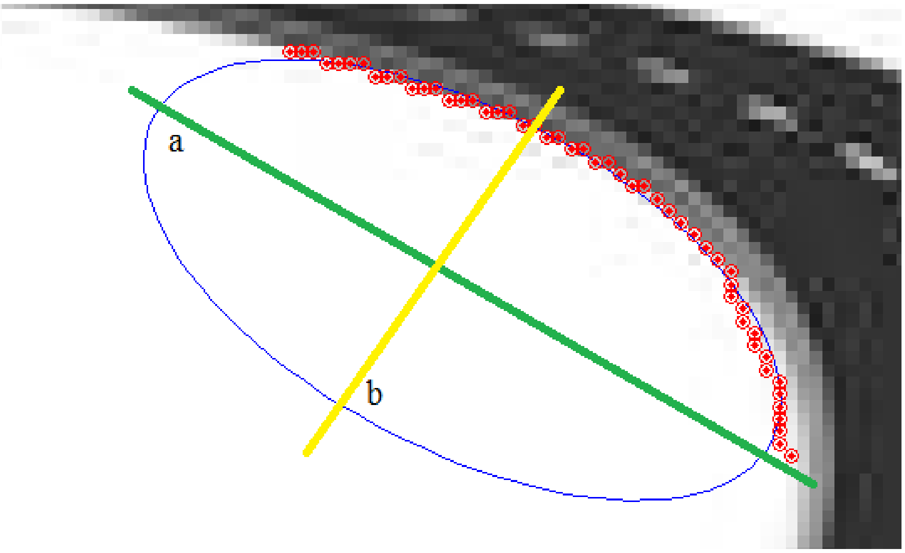
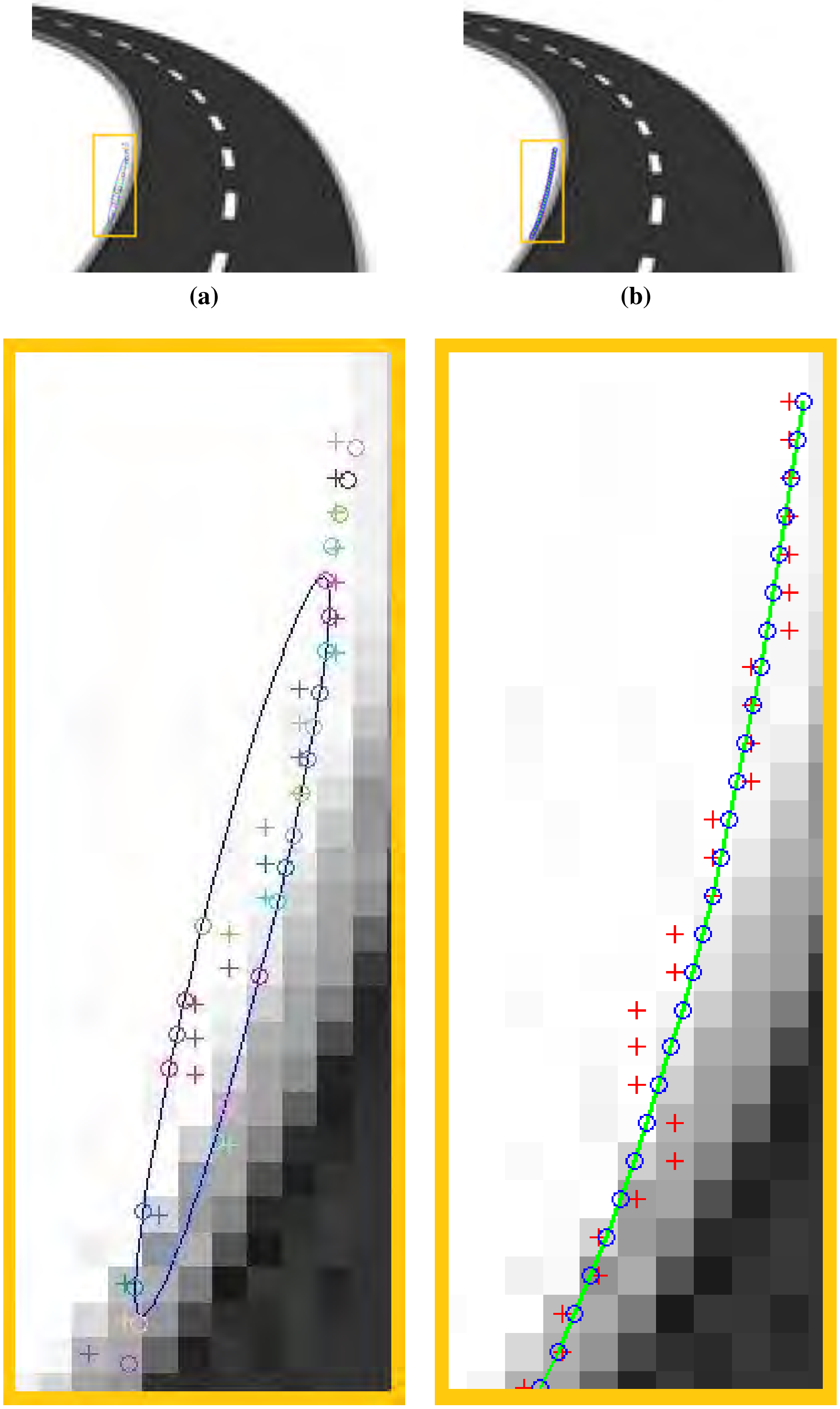


**Figure 3.24** Good ellipse fit with lower eccentricity.

Ellipse fitting gives reasonable output when the data points are positioned in such a way that fitted ellipse has lower ratio between major and minor axis (eccentricity is lower as shown in Figure 3.24). However, when data points are too flat that means fitted curvature is very low, ellipse fitting is not appropriate for our application. Much the elongated shaped ellipse produced by the ellipse fitting, further the distant from our ground truth fitting. $d_{\text{curvature}}$, and $d_{\text{gap}}$ will be high for ellipse fitting shown in Figure 3.25 (a). Let, **o** are pixels belong to $c \cup c'$. Corresponding ellipse pixel for each original pixel is shown with same colored **+**. Here, **o** are spread all over the ellipse and have higher discontinuity. The fit error is also large as most of the **o** is distant from its corresponding **+** compared to ground truth fit. Moreover, it generates the ellipse with larger eccentricity which makes the curve curvature very high near to two end points of the major axis. Therefore, ellipse fitting gives large dissimilarity for clusters $c$ and $c'$. However, the ground truth fitting (Figure 3.25 (b)) shows that they have lower dissimilarity



(a)

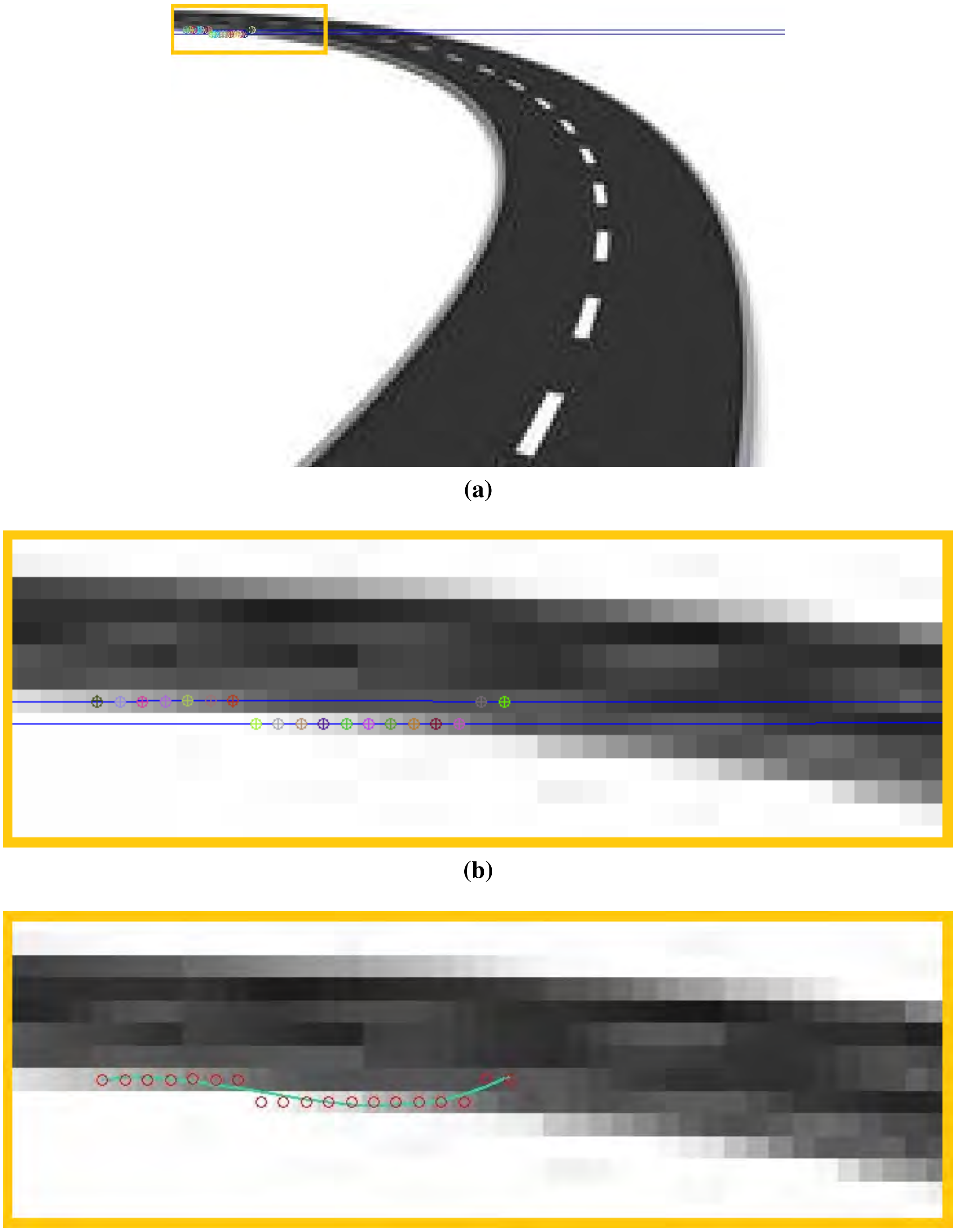
(b)

(c)

(d)

**Figure 3.25** As the eccentricity tends toward 1, the fitted ellipse gets more apart from ground truth , (a) ellipse fitting, (b) ground truth fitting, (c) zoom view of orange colored rectangular area of (a), and (d) zoom view of orange colored rectangular area of (b).

and good candidates to be merged for next level of the hierarchy tree.



**(a)**

**(b)**

**(c)**

**Figure 3.26** Bad ellipse fit with data places in parallel structure, (a) ellipse fitting , (b) zoom view of orange colored rectangular area of (a), and (c) ground truth fitting of that area.

Figure 3.26 (a) shows a case where members of $c \cup c'$ form a parallel structure. One set of data is parallel to other set. In such situation, although fit error is very small but ellipse fitting perform worst with extremely high curve discontinuity and high curvature at both ends of the major axis. Ground truth fitting is shown in Figure 3.26 (b) which verifies that $c$ and $c'$ are good candidates to be merged with very small fit error and discontinuity.

### 3.6.2 Polynomial fitting and its problems

A polynomial of degree $n$ is a function of the form

$$f(x) = a_n x^n + a_{n-1} x^{n-1} + a_{n-2} x^{n-2} \cdots a_1 x + a_0 \tag{3.18}$$

where, each coefficient $a_k \in \mathbb{R}$ is a real number, $a_n \neq 0$, and $n$ is a non-negative integer.

**Fitting problem with polynomial**

Although polynomial covers lots of curve shapes, however some shapes cannot be fitted properly by it. Figure 3.27 (a) shows a case where polynomials with degree 2 to 3 are poorly fitted. Here, red and magenta colored dots are members of 2 clusters for which need to calculate the dissimilarity by fitting the polynomial. Larger the fit error, greater the dissimilarity. Figure (b) shows the ground truth fit to these clusters, which is far away from original polynomial fit for both $2^{nd}$ and $3^{rd}$ degree polynomials.

### 3.6.3 Polynomial fitting with PCA

Principal component analysis (PCA) is a well-established technique for data analysis, compression, pattern recognition, and specially dimensionality reduction. It reduces the higher dimensional data by detecting the directions of the maximal variation of the data, called principal components. First principle component is the direction along which the data vary most. This direction is found by constructing a linear trend line in such a way that it minimizes the sum of the squared distances from the points to the line shown in Figure 3.28.

Our approach uses PCA to get the direction along the maximum variation of the data (Fig-

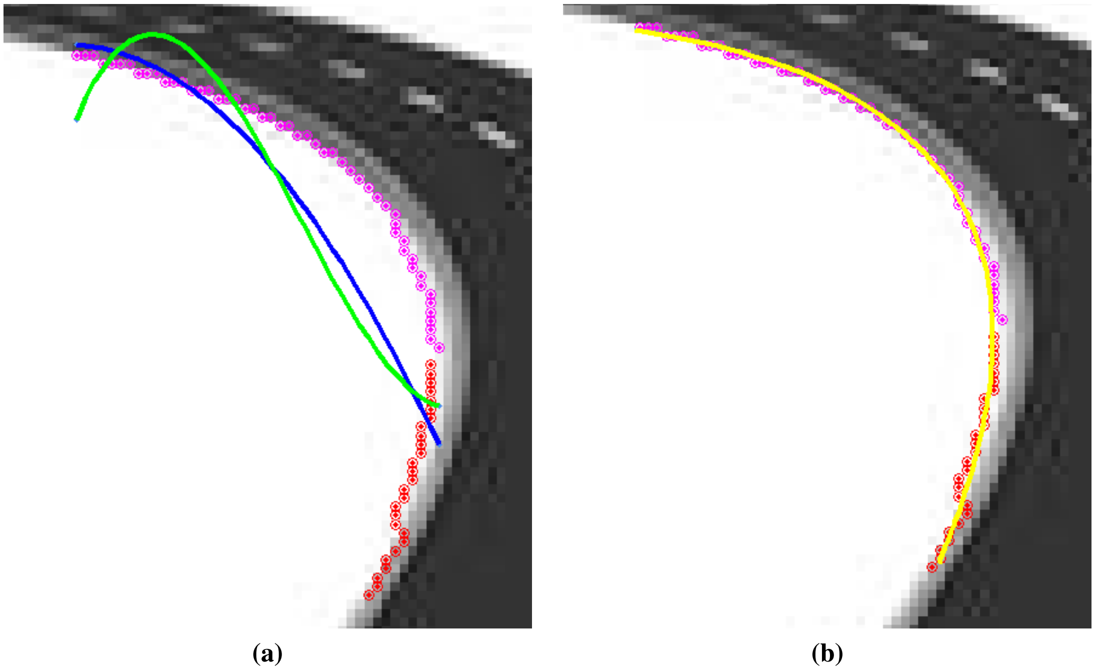


**Figure 3.27** Polynomial fitting problem, (a) $2^{nd}$ (blue colored curve) and $3^{rd}$ (green colored curve) degree polynomial fit, (b) ground truth fit.

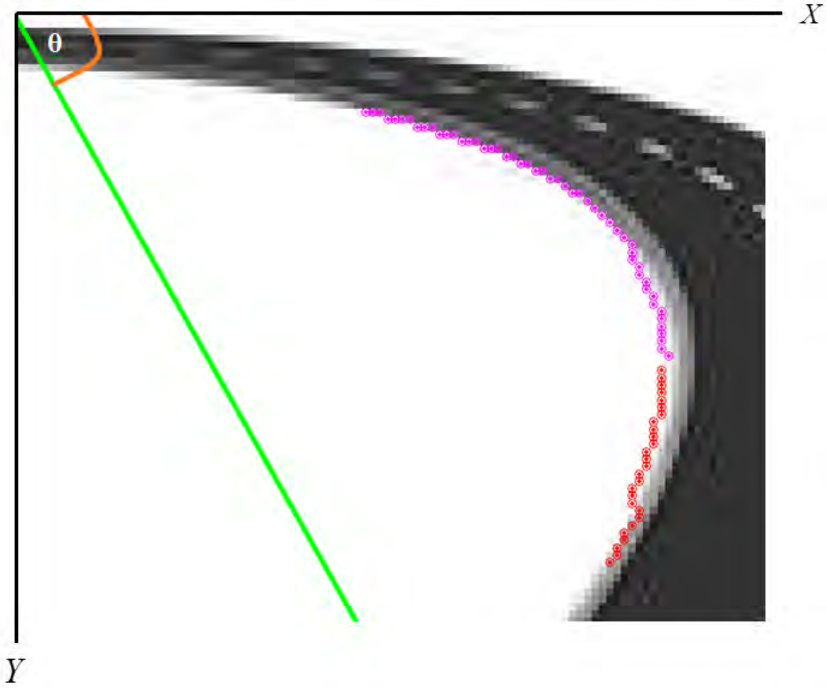
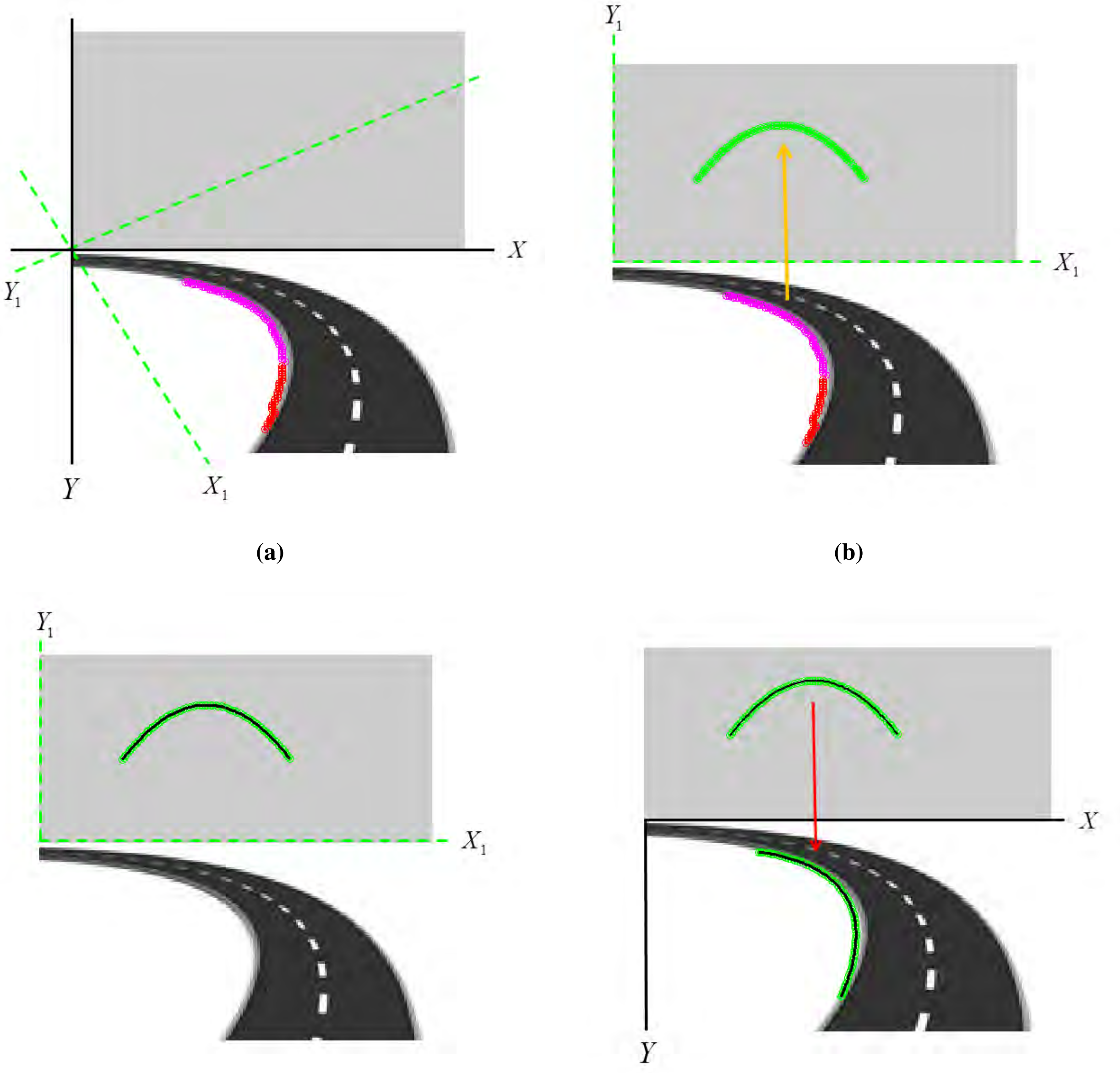


**Figure 3.28** First principle component from PCA gives the directions of the maximal variation (green color line) of the data shown by red and magenta colored dots.

ure 3.28). First, it creates new axis based on that direction ($\theta$) (Figure 3.29 (a)) and rotates the data (Figure 3.29 (b)). In this step. data are actually moved from original axis to the new axis with angle $\theta$ generated in PCA. Then, fit the polynomial to those rotated data (black curve in Figure (c)) and compute the dissimilarity. At that time, the fitted polynomial curve will be more close to the fitted ground truth curve as the data form the shape that a polynomial can handle more efficiently. Finally, the data are rotated back to the original axis (Figure (d)) which is more closer to the ground truth fit shown in Figure 3.27 (b).



**(a)** **(b)**

**(c)** **(d)**

**Figure 3.29** Polynomial fit with PCA, (a) new axis drawn from the direction ($\theta$) given by PCA, shown as green color dashed line, (b) data rotated to the new axis, (c) fit poly to the rotated data, and (d) fitted data rotated back to the original axis.

### 3.6.4 Choice on degree of polynomial

Overfitting to the data means fit the data accurately which mislead the intention of fitting for particular application. Overfitting generally happens when a model is extremely complex, such

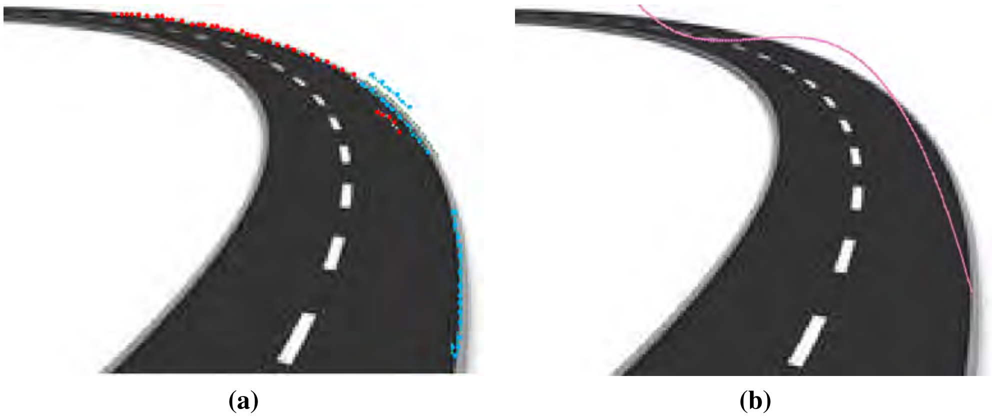
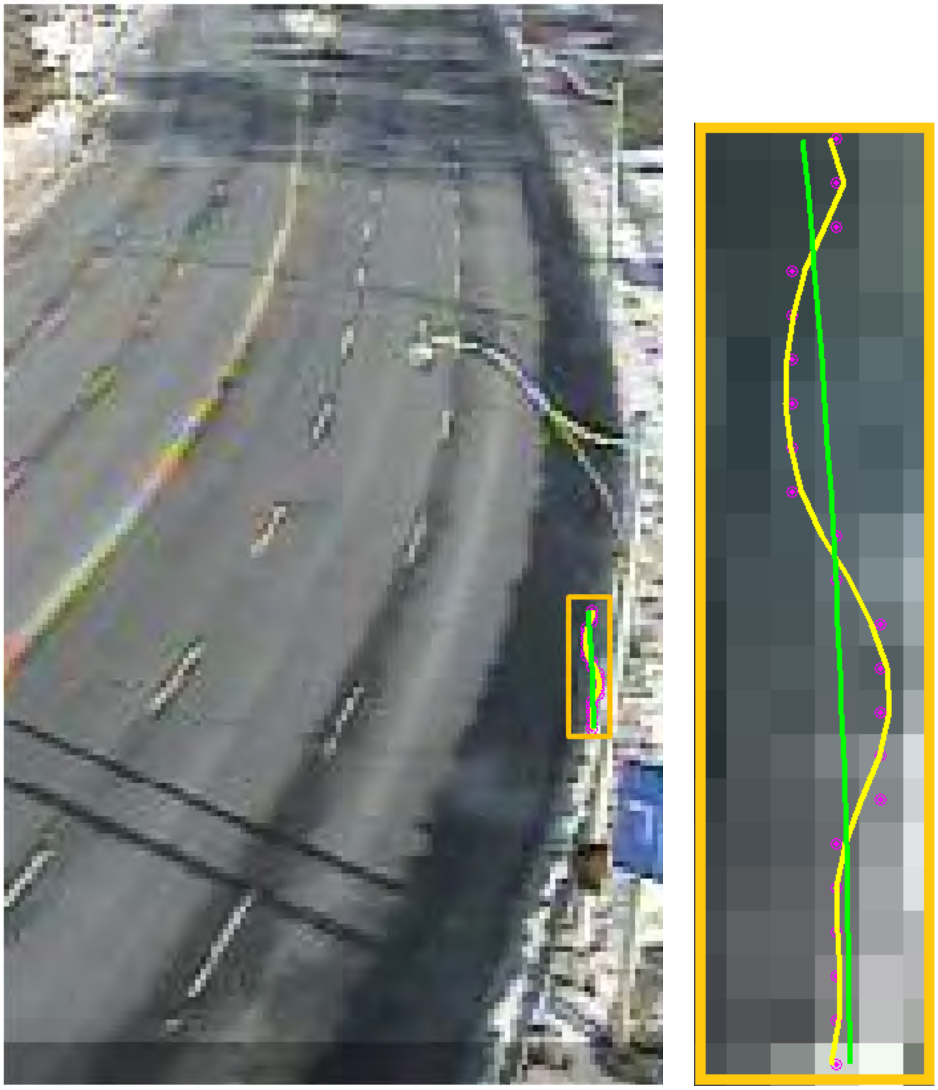


**Figure 3.30** Data overfitting using third degree polynomial, (a) two clusters shown by red and cyan colors and (b) third degree polynomial fitting (shown as purple colored curve) leads to wrong path because of data overfitting.

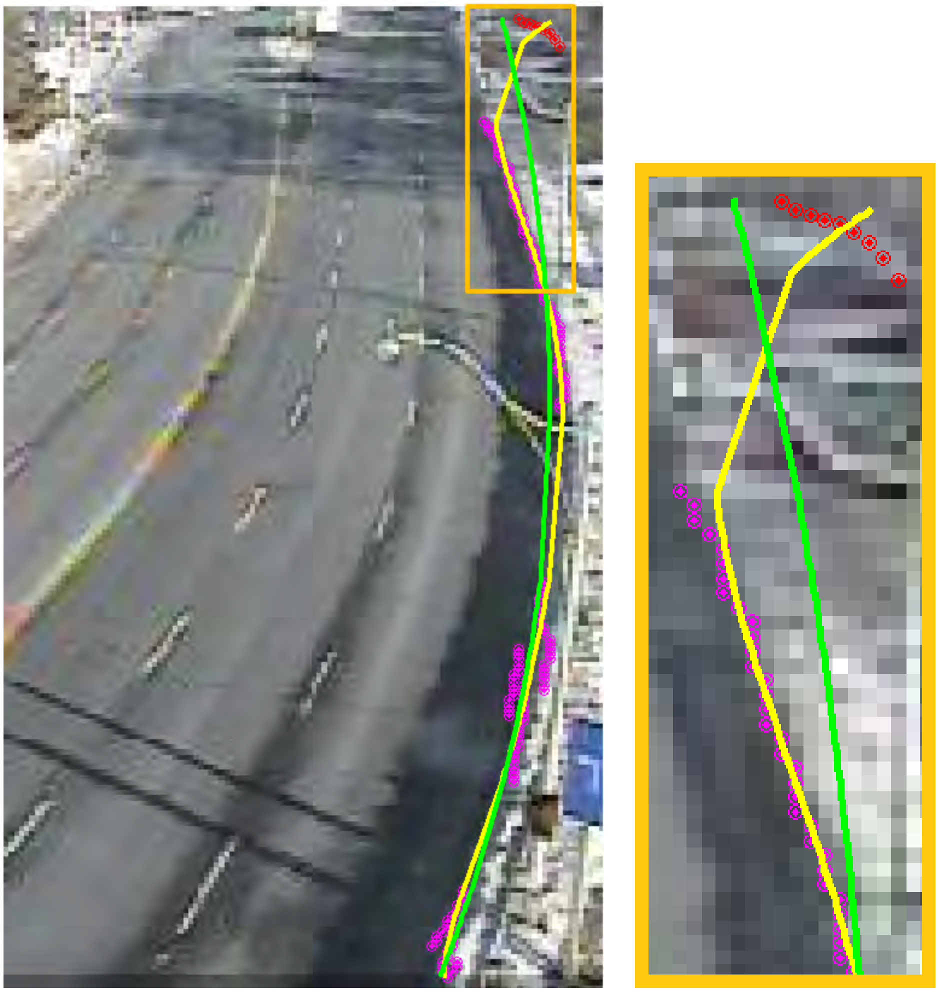
as having too many parameters or when the data is too noisy. In our application, the shape of most of the road boundaries is second degree polynomial. However, because of noise and lower quality of the road image, higher degree polynomial sometime fits better than lower degree. In such case fit error is small but not always appropriate for our application. The quality of our traffic images is low and has lots of unwanted data on account of weather and lighting challenges. Those generate substantial amount of noise. Therefore, good fitting to each of the data does not always lead to better outcome for the application.

Figure 3.30 (b) shows $3^{rd}$ degree polynomial fitting which identify boundary as spline because of having more parameters to satisfy than $2^{nd}$ degree polynomial. Although, the ground truth boundary will be a second degree curve. Figure 3.31 shows $2^{nd}$ degree and $10^{th}$ degree polynomial fit to the data points by green and yellow colored curves respectively. Although higher degree polynomial fits better than $2^{nd}$ degree, however shape of the fitted curve for $2^{nd}$ degree matches more toward the real boundary of the road. In Figure 3.32, shape of the fitted curve (yellow curve) for higher degree ($6^{th}$ degree) polynomial lost to become close as original road boundary because of the noisy data shown in red colored dot. Conversely, $2^{nd}$ degree curve (green curve) stay close to the ground truth road boundary as for its simplicity and having less parameter to satisfy.



(a) (b)

**Figure 3.31** Data overfitting with, (a) polynomial of degree 10 (yellow colored curve) compared to degree 2 (green colored curve) and (b) zoomed view of the orange colored rectangular area .



(a) (b)

**Figure 3.32** Data overfitting, (a) higher degree polynomial fitting (shown as yellow colored curve) leads to wrong path because of data overfitting and (b) zoomed view of the orange colored rectangular area.

# Chapter 4

# Experimental Results

The proposed technique is able to find out the curved road boundary by analysing a single traffic image. Data set, a collection of images containing curved road taken from the cameras installed along the Ontario 401 highway has been experimented. We allow manually labeled image frame at each location to serve as ground truth for that particular location. While some of these images are collected during day time, the rest of them are collected during night time. We also consider cameras those installed at locations where the highway is only visible through the vehicles highlights at night. We consider a wide range of brightness, color, texture, illumination, shadow, and surrounding environmental variation in the data set. We choose frame resolution as $320 \times 240$. Thus, we aim to demonstrate the robustness of the proposed road boundary detection technique under resource constrained setups using these low-resolution images.

Figure 4.1 and 4.2 show a few outputs of our technique in different locations. At each scene, the first row represents the ground truth and the second row shows our top ranked pair which is our final output. The results show that our method is capable to handle a wide-variety of challenging lighting and environmental conditions.

Figure 4.1 shows several road boundaries generated at day and night time using our approach. Here, Figure 4.1 (a) and (b) have extreme lighting problem at day time. (c) and (d) are

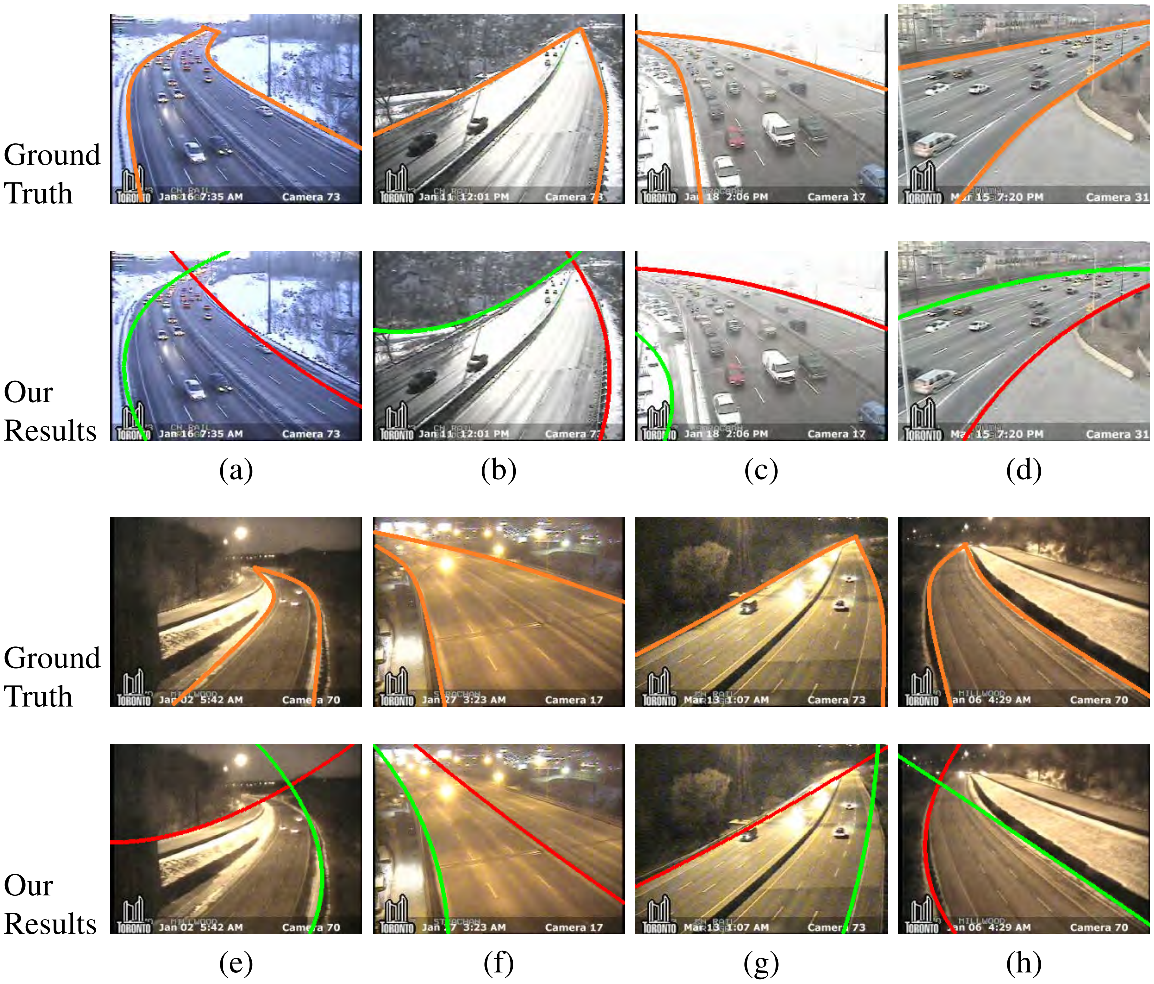


**Figure 4.1** Results of the proposed technique in 8 different scenes at day and night lighting. The first row in each scene is the ground truth, the second row is our results.

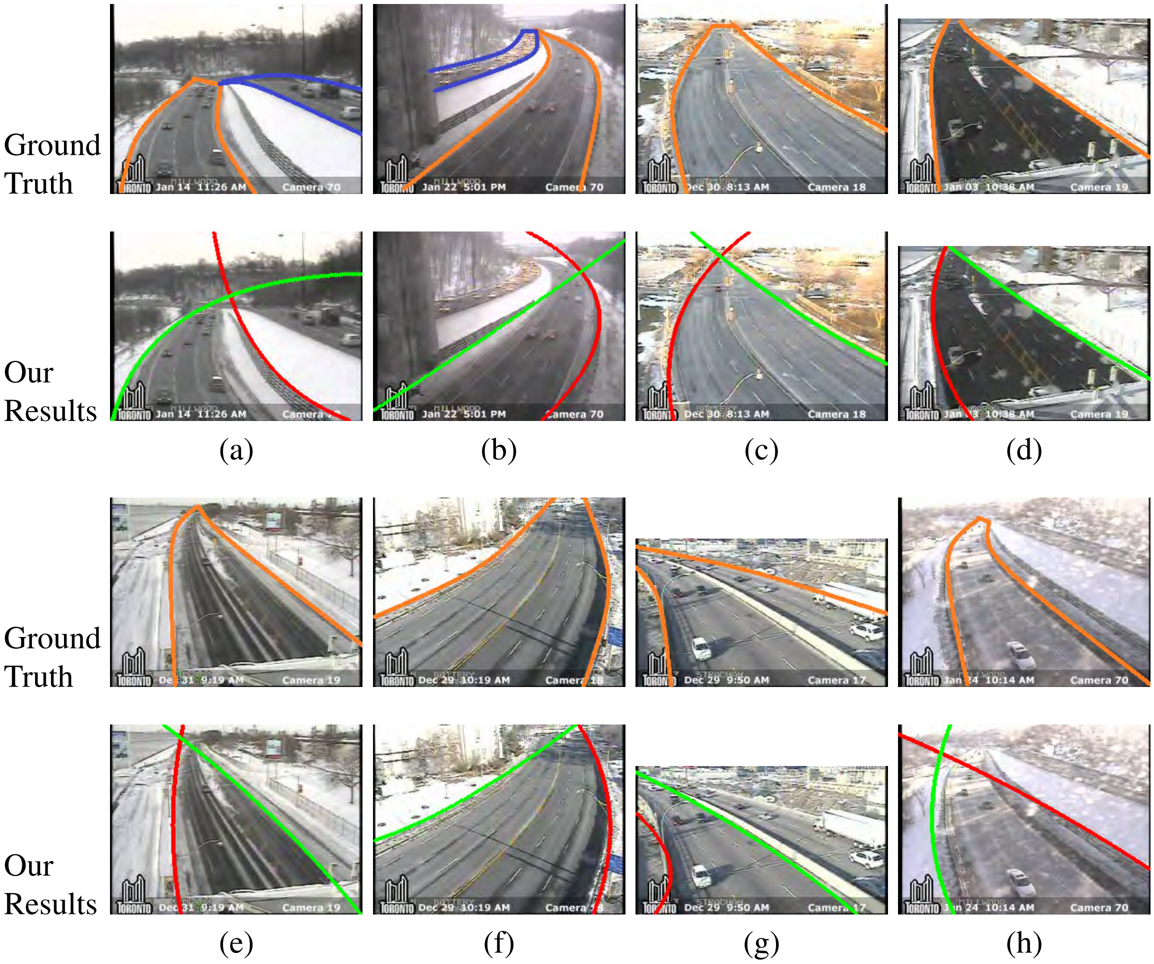


**Figure 4.2** Results of the proposed technique in 8 different scenes containing multiple roads (a) and (b), boundary discontinuity (c), snow (d) and (e), shadows (f) to (h). The first row in each scene is the ground truth and the second is our results.

example of showing road with lots of vehicles. While (e) and (f) demonstrate the lighting problem by lamp post at night time. (g) at night is a good example that handles vehicle headlight. (h) is an example of detecting dominant road boundary between two roads.

Figure 4.2 (a) and (b) consist of two curved roads. Here dominant road boundary is detected. (c) handles road discontinuity caused by another small side road. In (d) road is unclear because of snowing. (e) shows a case where the snow places on the road region. Shadows generated by car, tree etc. are presenting in Figure 4.2 (f) to (h). In (h), dominant road is detected between two roads.

We evaluate the performance of the proposed approach by varying three important parameters of our algorithm.

**Threshold** is a parameter that controls the number of edge pixels produced by edge detection;

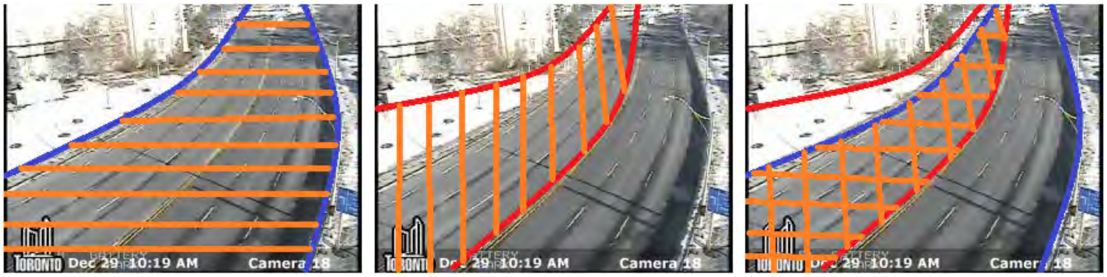
**Minimal number of edge pixels** for edge linking that controls the presence of noise; and

**Top candidates** selection from the hierarchical clustering tree for pairwise analysis which controls the possible number of pairs to find final road boundary.

## Performance evaluation by precision and recall

The precision and recall comparison method [5] have been plotted for performance evaluation of our proposed technique as shown in Figure 4.3. Our ground truth road boundary is human labelled. We define the precision as the ratio of the intersected area by the estimated road boundary and the ground truth to the area of the estimated boundary. On the other hand, we define recall as the ratio of the intersected area by the estimated boundary and the ground truth to the area of the estimated truth.

Martin *et al.* [22] assigns each pixel a probability value between 0 to 1, which is the probability of that pixel to be an edge pixel. Thresholding is then applied to filter out some pixels as non-edged pixels. Therefore, selecting pixel scale evidences highly depends on that


TORONTO Dec 29 10:19 AM Camera 18
Dec 29 10:19 AM Camera 18
Dec 29 10:19 AM Camera 18

(a) (b) (c)

**Figure 4.3** Precision and recall calculation (a) human labelled ground truth road region by two boundaries (b) identified region by proposed approach, and (c) correctly identified region.

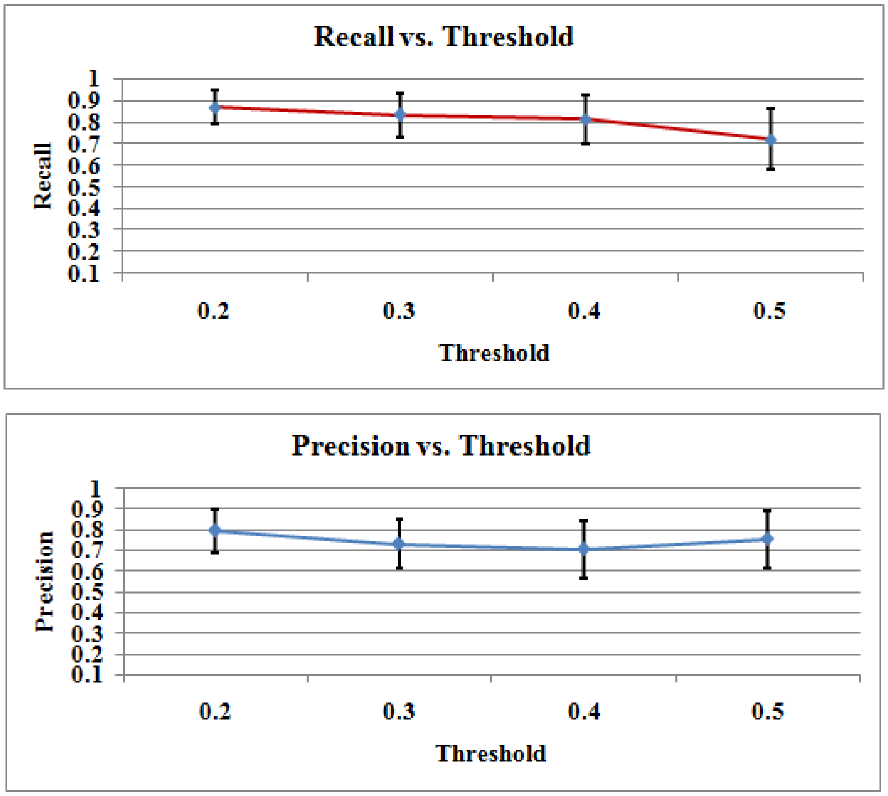


**Figure 4.4** Precision and recall graph by varying threshold of edge pixels.

threshold value which also affects clustering and as well as final road boundary detection. Figure 4.4 shows the effect of thresholding by varies its value from 0.2 to 0.5. Here, it is visible that as the threshold value increases, the precision and recall decrease from 80% to 75% and from 87% to 72% respectively. Usually the higher the threshold, the lower the number of pixel scale evidences which may leave a lot of edge pixels those belong to original road

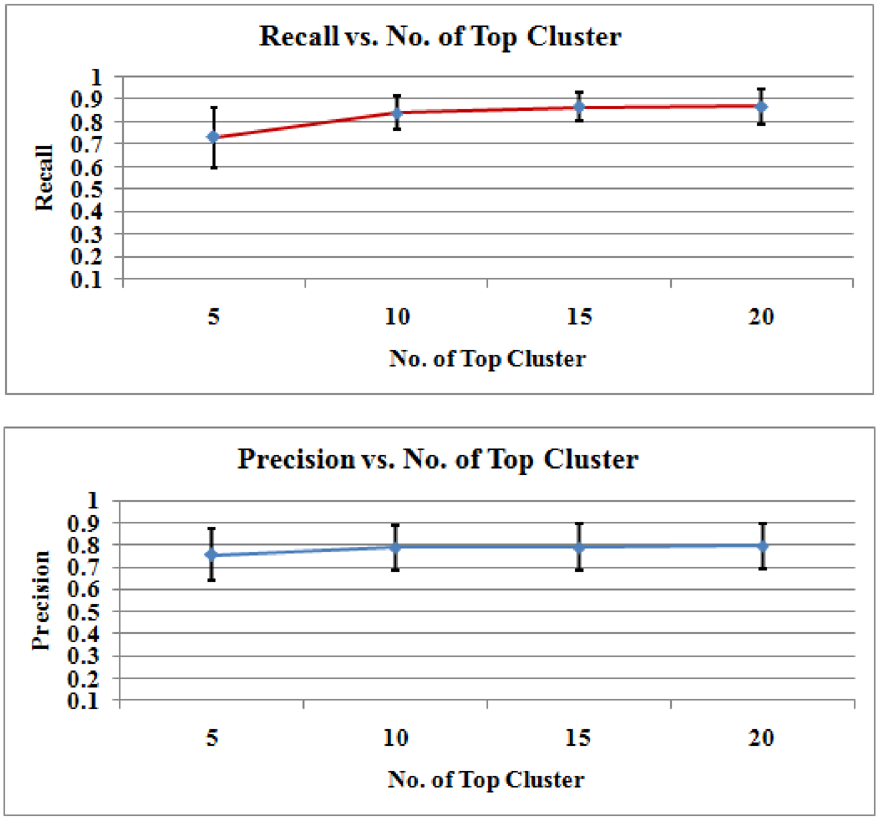


**Figure 4.5** Precision and recall graph by varying the no. of top clusters.

boundary. Therefore the accuracy falls down. Moreover, it has higher standard deviations at higher threshold level (0.4, 0.5) than lower threshold level (0.3, 0.2) which also implies inconsistent performance over the data set.

Top $k$ clusters are paired with each other and ranked to find the final top pair as road boundary, described in Section 3.5. The accuracy of the application also varies based on the selection of the number $k$. In Figure 4.5, both the precision and recall increase with the increasing value of top $k$. The chance of being the original road boundary is in the top $k$ cluster goes higher with larger number of $k$. Therefore, the precision and recall rises from $76\%$ to $80\%$ and from $73\%$ to $87\%$ respectively for the number of top clusters 5 to 20.

Intermediate candidates are generated by adding neighbouring edge pixels. The minimum number of pixel scale candidates for building intermediate candidates depends on the setting of minimum length of the segment joined by *edge linking*. Small length allows more noisy can-

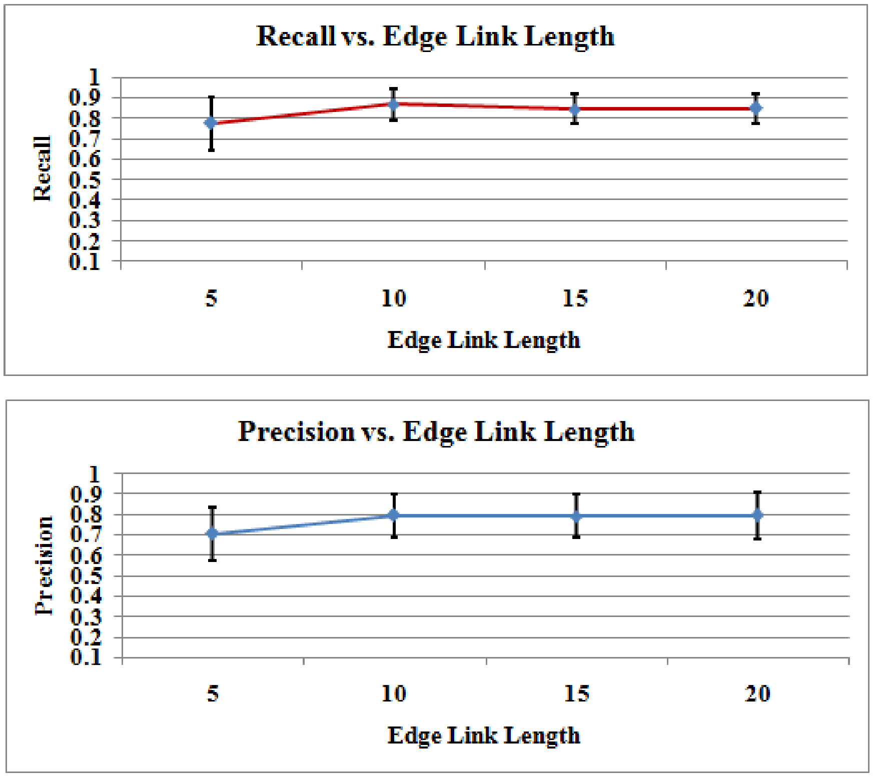


**Figure 4.6** Precision and recall graph by varying minimum edge link length.

didates to incorporate intermediate candidates. Precision and recall graph of Figure 4.6 shows how does it affect to our application. We can see when length is 5, the precision and recall both are low. Moreover, the standard deviation is high to indicate its inconsistent performance. While in other minimum edge link lengths like 10, 15, 20 they show better performance with higher precision and recall with stable standard deviation.

## Performance evaluation by run time

Figure 4.7 shows the run time (in sec.) with respect to the threshold for selecting pixel scale evidences considering different number of top clusters. Since increasing threshold implies decreasing number of pixel scale evidences, therefore run time also decreases. Different colored series represent this effect for number of top clusters from 10 to 25.

Figure 4.8 shows how the run time varies with the increase of number of top clusters with

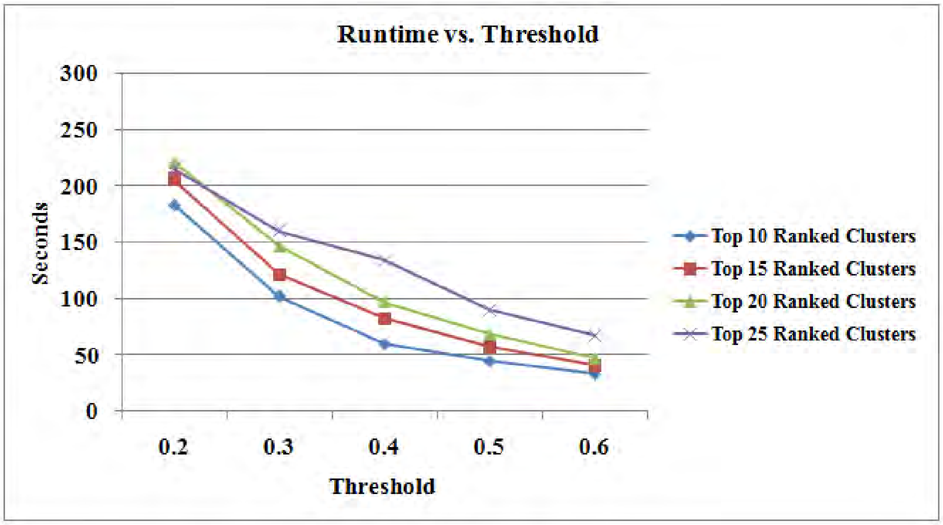


**Figure 4.7** Run time (in second) by varying threshold in different no. of top clusters.

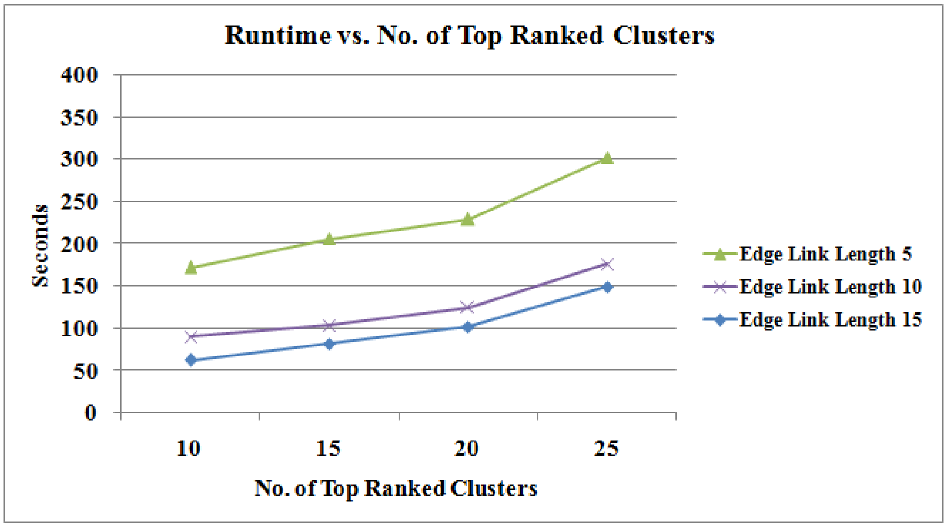


**Figure 4.8** Run time (in second) by varying the no. of top clusters in different minimum edge link length.

minimum edge link length. Because of performing pairwise ranking between each of the top clusters, runtime differs highly. Therefore, in all cases (minimum edge link length from 5 to 15) run time is almost three times for the number of top clusters 25 than for the number of top clusters 5.

Figure 4.9 shows the run time as a function of threshold value for filtering pixel scale evidences with the variation of the minimum edge link length. Runtime is worse at threshold

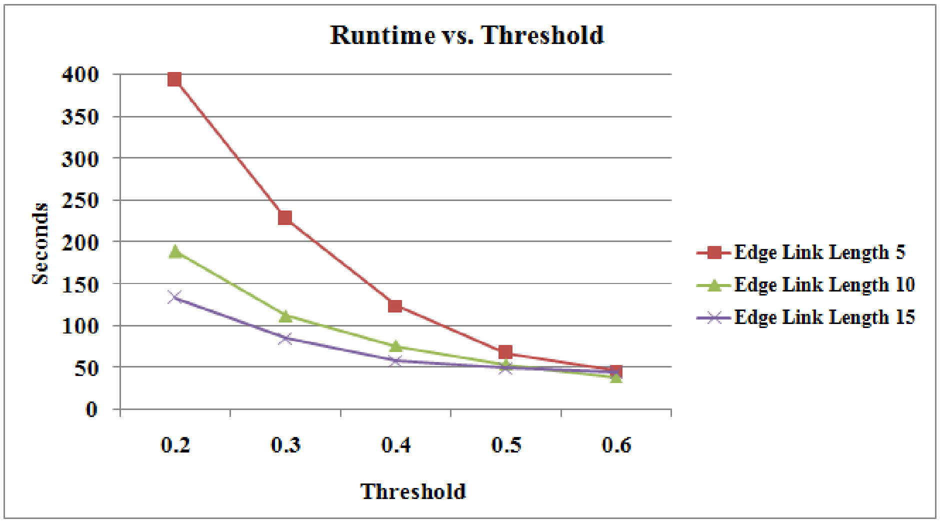


**Figure 4.9** Run time (in second) by varying threshold in different minimum edge link length.

0.2 with the minimum edge link length 5.

## Boundary detection on straight roads

The proposed algorithm is also examined on straight road obtained from Helala *et al.* data set to find dominant straight road boundary. This data set includes the traffic images of straight roads along Ontario 401 highway . We achieved $73\%$ precision and $76\%$ recall for that data set (considering top 20 clusters, 0.2 threshold on edge pixel and minimum 10 pixels length of *edge linking*). Figure 4.10 shows some results of our technique for straight road in different locations. At each scene, the first row represents the ground truth and the second row shows the top ranked pair which is our final output. The results show that the proposed approach is also capable to handle straight road at both day and night time.

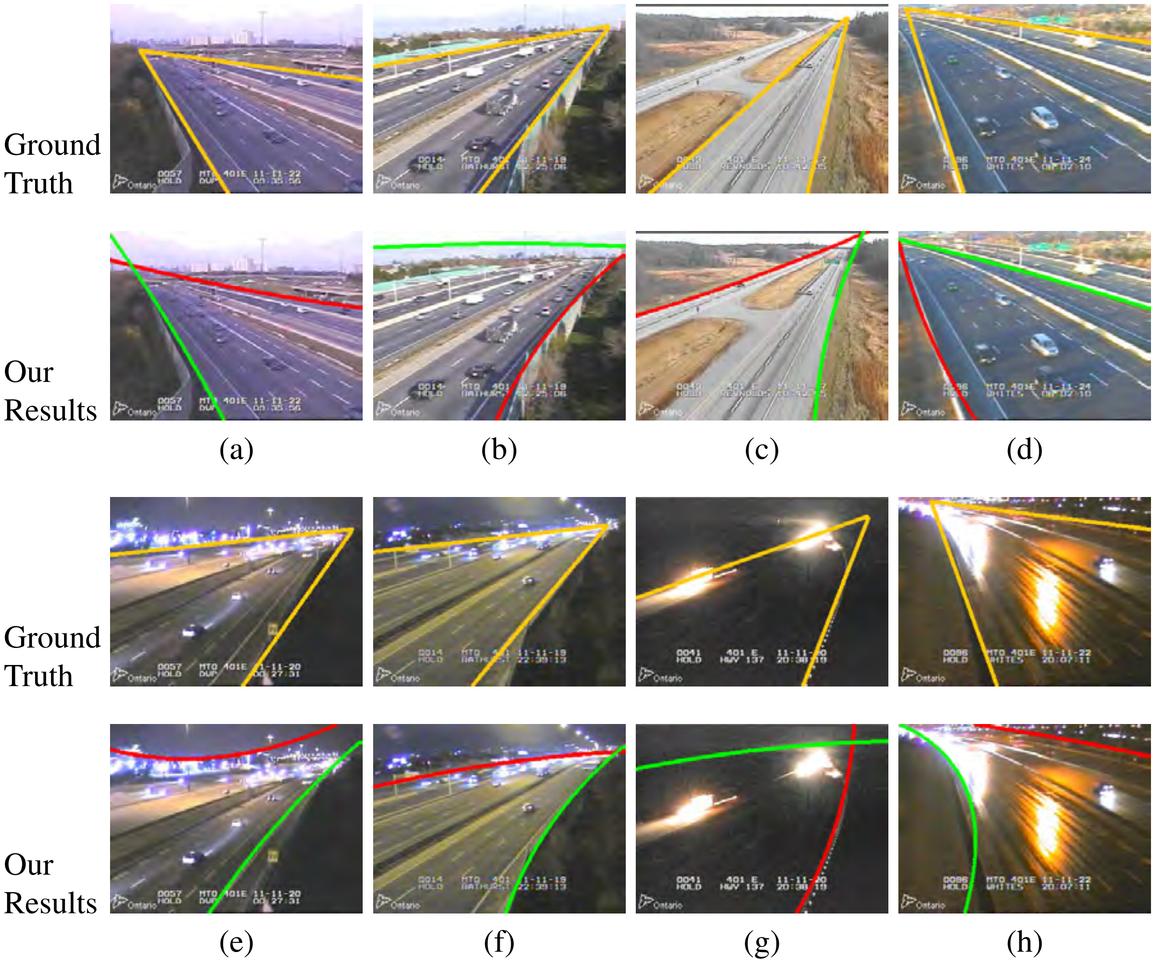


**Figure 4.10** Results of the proposed technique on straight road in 8 different scenes at day and night lighting (from Helala *et al.* data set [11]). The first row in each scene is the ground truth, the second row is our results.

# Chapter 5

# Conclusions and Future Work

This thesis develops a dominant curved road boundary identification algorithm, which is able to identify road boundaries in images captured from traffic cameras. These images exhibit a variety of environmental and lighting conditions. The proposed algorithm is especially suitable for large scale traffic camera setups for the following two reasons:

- the proposed algorithm needs only low-resolution images, saving bandwidth resources that are at a premium in large scale traffic camera networks; and
- the proposed algorithm is able to identify road boundaries given a single image, again conserving network resources.

The proposed method begins by identifying edge pixels in the traffic image. The edge pixels are then grouped together to construct edge segments via *edge linking*. The edge segments represent the bottom-level of a hierarchical clustering pyramid. At each stage of the hierarchical clustering the two edge segments that are closest to each other according to the metric that we describe in Chapter 3 are merged to form a larger edge segment. Each node in the hierarchical clustering tree is ranked according to the above mentioned metric and the top ranked nodes are paired together to construct road boundary hypotheses. Each node pair (or road boundary hypothesis) is evaluated using appearance and perspective cues available from the image and the top ranked node pair is returned as the dominant road boundary.

We have evaluated the proposed approach on a real data set collected over many months from traffic cameras installed on the section of Hwy 401 that passes through the Greater Toronto Area (GTA). The proposed method is able to correctly identify road boundaries in challenging traffic images that exhibit a variety of lighting and environmental conditions. The work presented here builds upon prior work by Helala *et al.* [11]. Their work was restricted to straight road boundaries and relied upon the ability to compute vehicular motion to differentiate between road and non-road regions. The work presented here addresses two limitations of their work. Firstly, our method can deal with curved road boundaries. Secondly, the proposed method needs only one image to detect road boundaries.

We can think of five limitations of the current work: 1) The proposed scheme can only identify dominant road boundaries. Presently it is unable to identify lanes or boundaries of multiple roads seen in the image. 2) To date we have not compared the performance of our method to that of other existing techniques. Such comparisons are of course important and we leave it as a possible future direction. 3) We have implemented and evaluated the proposed algorithm within the Matlab development environment. The current implementation cannot be used for real-time data processing. 4) Lastly, the current technique simply ranks possible road boundaries but does not assign scores to these boundaries. This suggests that another system that relies upon our method to detect road boundaries has no way of knowing the quality of the boundaries returned by our method. 5) We model road boundaries as quadratic polynomials. This assumption is suitable for our data set. However, it also means that the proposed method cannot deal with S-shaped roads. Furthermore, we also make an assumption that the road boundaries exhibit low to medium curvature, which suggests that the proposed method might fair poorly on U-shaped roads. The future work will focus on one or many of the items listed above.

# Bibliography

[1] Traffic congestion management and traffic signal coordination, 2013. Retrived from http://app.toronto.ca/tmmis/viewAgendaItemHistory.do?item=2013.PW23.11/ (last accessed on 30 October 2013).

[2] Traffic reporting and monitoring, 2013. Retrived from http://www.cp24.com/traffic/traffic-alert/ (last accessed on 30 October 2013).

[3] Y. Alon, A. Ferencz, and A. Shashua. Off-road path following using region classification and geometric projection constraints. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 689–696, 2006.

[4] M. Aly. Real time detection of lane markers in urban streets. In *Proc. IEEE Intelligent Vehicles Symposium*, pages 7–12, Eindhoven, The Netherlands, June 2008.

[5] R. Baeza-Yates and B. Ribeiro-Neto. Modern information retrieval. New York: ACM Press, Addison-Wesley, 1999.

[6] A. Borkar, M. Hayes, and M. Smith. Robust lane detection and tracking with ransac and kalman filter. In *Proc. 16th IEEE International Conference on Image Processing (ICIP)*, pages 3261–3264, 2009.

[7] B. Coifman, D. Beymer, P. McLauchlan, and J. Malik. A real-time computer vision system for vehicle tracking and traffic surveillance. *Transportation Research Part C: Emerging Technologies*, 6(4):271 – 288, 1998.

[8] R. T. Collins and R. S. Weiss. Vanishing point calculation as a statistical inference on the unit sphere. In *Proc. Third International Conference on Computer Vision*, pages 400–403, 1990.

[9] R. Danescu and S. Nedevschi. Probabilistic lane tracking in difficult road scenarios using stereovision. *IEEE Transactions on Intelligent Transportation Systems*, 10(2):272–282, 2009.

[10] David Heckerman. A tutorial on learning with bayesian networks. Technical report, Learning in Graphical Models, 1996.

[11] M. A. Helala, K. Q. Pu, and F. Z. Qureshi. Road boundary detection in challenging scenarios. In *Proc. Ninth IEEE International Conference on Advanced Video and Signal-Based Surveillance*, AVSS '12, 2012.

[12] A.S. Huang, D. Moore David, M. Antone, E. Olson, and S. Teller. Finding multiple lanes in urban road networks with vision and lidar. *Autonomous Robots*, 26(2-3):103–122, 2009.

[13] R. Jiang, R. Klette, T. Vaudrey, and S. Wang. Lane detection and tracking using a new lane model and distance transform. *Machine Vision and Applications*, 22(4):721–737, July 2011.

[14] I. Katramados, S. Crumpler, and T. Breckon. Real-time traversable surface detection by colour space fusion and temporal analysis. In *Proc. 7th International Conference on Computer Vision Systems*, ICVS '09, pages 265–274, Berlin, Heidelberg, 2009. Springer-Verlag.

[15] Z. Kim. Robust lane detection and tracking in challenging scenarios. *IEEE Transactions on Intelligent Transportation Systems*, 9(1):16–26, 2008.

[16] H. Kong, J. Audibert, and J. Ponce. Vanishing point detection for road detection. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 96–103, Miami, FL, June 2009.

[17] H. Kong, J. Audibert, and J. Ponce. Vanishing point detection for road detection. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 96–103, 2009.

[18] P. Kovesi. Matlab and octave functions for computer vision and image processing, 2008. Retrived from `http://www.csse.uwa.edu.au/~pk/research/matlabfns/` (last accessed on 30 October 2013).

[19] R. Labayrade, J. Douret, J. Laneurit, and R. Chapuis. A reliable and robust lane detection system based on the parallel use of three algorithms for driving safety assistance. *IEICE Transactions*, E89-D(7):2092–2100, July 2006.

[20] J. Lou, Q. Liu, T. Tan, and W. Hu. Semantic interpretation of object activities in a surveillance system. In *Proc. International Conference on Pattern Recognition*, pages 777–780, 2002.

[21] P. C. Mali, B. B. Chaudhuri, and D. Dutta Majumder. Performance bound of walsh-hadamard transform for feature selection and compression and some related fast algorithms. *Pattern Recognition Letters*, 2(1):5 – 12, 1983.

[22] D. R. Martin, C. C. Fowlkes, and J. Malik. Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26:530–549, 2004.

[23] J.C. McCall and M.M. Trivedi. Video-based lane estimation and tracking for driver assistance: survey, system, and evaluation. *IEEE Transactions on Intelligent Transportation Systems*, 7(1):20–37, 2006.

[24] J. Melo, A. Naftel, A. Bernardino, and J. Santos-Victor. Detection and classification of highway lanes using vehicle motion trajectories. *IEEE Transactions on Intelligent Transportation Systems*, 7(2):188–200, June 2006.

[25] A. Nefian and G. Bradski. Detection of drivable corridors for off-road autonomous navigation. In *Proc. IEEE International Conference on Image Processing*, pages 3025–3028, 2006.

[26] M. Nieto, L. Salgado, F. Jaureguizar, and J. Arrospide. Robust multiple lane road modeling based on perspective analysis. In *Proc. 15th IEEE International Conference on Image Processing*, pages 2396–2399, 2008.

[27] D. Pomerleau. Ralph: rapidly adapting lateral position handler. In *Proc. IEEE Symposium on Intelligent Vehicles*, pages 506–511, 1995.

[28] C. Rasmussen and T. Korah. On-vehicle and aerial texture analysis for vision-based desert road following. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '05, pages 66–66, 2005.

[29] Y. Rubner and C. Tomasi. Coalescing texture descriptors. In *ARPA Image Understanding Workshop*, pages 927–935, 1996.

[30] F. Samadzadegan, A. Sarafraz, and M. Tabibi. Automatic lane detection in image sequences for vision based navigation purposes. *ISPRS Image Engineering and Vision Metrology*, 2006.

[31] H. Sawano and M. Okada. A road extraction method by an active contour model with inertia and differential features. *IEICE Transactions*, E89-D(7):2257–2267, July 2006.

[32] A. Soumelidis, G. Kovacs, J. Bokor, P. Gaspar, L. Palkovics, and L. Gianone. Automatic detection of the lane departure of vehicles. In *Proc. 8th IFAC Symposium on Transportation Systems*, pages 1045–50, Chania, Greece, June 1997.

[33] B. Stewart, I. Reading, M. Thomson, T. Binnie, K. Dickinson, and C. Wan. Adaptive lane finding in road traffic image analysis. In *Proc. 7th IEEE International Conference on Road Traffic Monitoring and Control*, pages 133–136, Napier University Edinburgh, Apr. 1994.

[34] Y. Wang, E. Teoh, and D. Shen. Lane detection and tracking using b-snake. *Image and Vision Computing*, 22(4):269 – 280, 2004.

[35] Y. Wang, E. Khwang Teoh, and D. Shen. Lane detection and tracking using b-snake. *Image and Vision Computing*, 22(4):269–280, April 2004.

[36] S. Will, L. Hermes, J. M. Buhmann, and J. Puzicha. On learning texture edge detectors. In *Proc. of the International Conference on Image Processing*, pages 877–880, 2000.

[37] S. Wu, H. Chiang, J. Perng, C. Chen, B. Wu, and T. Lee. The heterogeneous systems integration design and implementation for lane keeping on a vehicle. *IEEE Transactions on Intelligent Transportation Systems*, 9(2):246–263, 2008.

[38] G. Zhang, N. Zheng, C. Cui, G. Yang, and Z. Yuan. An efficient road detection method in noisy urban environment. In *Proc. IEEE Symposium on Intelligent Vehicles*, pages 556–561, 2009.

[39] S. Zhou, J. Xi, J. Gong, G. Xiong, and H. Chen. A novel lane detection based on geometrical model and gabor filter. In *Proc. Intelligent Vehicles Symposium*, pages 59–64, San Diego, CA, June 2010.