BIG DATA ANALYTICS AND PROCESSING FOR URBAN SURVEILLANCE SYSTEMS

LING HU

School of Computing and Communications
Lancaster University

A thesis submitted for the degree of Doctor of Philosophy

October 2019

Declaration

I hereby confirm that the content presented in this thesis is totally original and have not been submitted in whole or in part for consideration for any other degree or qualification of any other university. Any information which derived from other sources has been indicated inside this thesis.

Ling Hu

October 2019

Acknowledgements

The author would like to acknowledge Professor Qiang Ni for his guidance and support throughout this work. Furthermore, the production of this thesis would not have been possible without the support from the Institute of Software Application Technology, Guangzhou & Chinese Academy of Sciences (CAS) and the Dongguan CAS Smart City Software Co. Ltd., China.

Abstract

Urban surveillance systems will be more demanding in the future towards smart city to improve the intelligence of cities. Big data analytics and processing for urban surveillance systems become increasingly important research areas because of infinite generation of massive data volumes all over the world.

This thesis focused on solving several challenging big data issues in urban surveillance systems.

First, we proposed several simple yet efficient video data recoding algorithms to be used in urban surveillance systems. The key idea is to record the important video frames when cutting the number of unimportant video frames.

Second, since the DCT based JPEG standard encounters problems such as block artifacts, we proposed a very simple but effective method which results in better quality than widely used filters while consuming much less computer CPU resources.

Third, we designed a novel filter to detect either the vehicle license plates or the vehicles from the images captured by the digital camera imaging sensors. We are the first to design this kind of filter to detect the vehicle/license plate objects.

Fourth, we proposed novel grate filter to identify whether there are objects in these images captured by the cameras. In this way the background images can be updated from time to time when no object is detected.

Finally, we combined image hash with our novel density scan method to solve the problem of retrieving similar duplicate images.

Table of Contents

CHAPTER 1

INTRODUCTION

1.1 Background

1.1.1 Motivation and Research Objectives

As we know, according to the 3V's definition, big data has three main characteristics, which are volume, velocity and variety [1]. Especially, the videos data is the biggest among all types of data. It is obvious that the research of videos data is the key point of big data research areas. According to Cisco's statistics, images and videos make up about 90 percent of overall IP traffic [2]. Hence video data analysis is receiving more and more attention due to its increasing applications and huge data volume. For instance, the data volume of all video surveillance devices in Shanghai, China is up to 1 TB every day [3]. The explosive increasing number of video resources has brought an urgent need to develop intelligent methods to reduce the pressure on the management processes [4].

The applications of urban surveillance systems include counter surveillance in banks, entrance monitoring of a safeguard region, traffic monitoring of crossroad, and so on. In most cases, the video cameras are fixed and the surveillance regions are also located constantly. In current urban video surveillance systems, there are huge amount of cameras working 24 hours per day, 7 days a week. Every camera focuses on a fixed area, in order to show the scenarios instantly, clearly and fluently. The frame per second (FPS) of every camera is normally kept at 30. In order to get higher quality for instant monitoring, some cameras increase the speed from 30 FPS to 60 FPS, or even higher (e.g. some cameras even reach 100 FPS). In these systems, digital video recorders (DVRs) are used to record the video frames; the most common number of inputs for a DVR is 1, 2, 4, 8, 16 and 32. Obviously the key components of the DVRs are huge hard discs (HDs). There are three key research challenges which motivate my PhD research work: 1) Although various compression techniques are currently used to reduce the size of every frame, the demand for high capacity HDs still increases rapidly due to huge amount of frames generated and to be generated explosively in the future. 2) As videos are composed of visual images, reducing the sizes of images while keeping a reasonable visual quality is an important research topic. However, the widely used JPEG image compression techniques have insurmountable problems. For instance, image encoding deficiencies such as block artifacts have to be removed frequently which demands research. 3) Intelligent vehicle license plates (or vehicles) recognition is a key component of Urban Surveillance Systems, with important applications, such as road traffic monitoring, security control of restricted areas, automatic parking lots access control, searching for stolen vehicles, etc. This is a challenging research topic given its high computational complexity.

Therefore, the main objectives of my PhD research are as follows:

- To develop low-complexity intelligent big video data recording algorithms to reduce the HDs' space.
- To reduce the visual block artifacts of JPEG Images by using/designing novel methods for urban surveillance systems.

- To design novel low-complexity intelligent vehicle license plate (or vehicle) recognition methods.

1.1.2 Our Research Topics

First, we research on low-complexity intelligent big video data recording algorithms to save the HDs' space. The second topic is to reduce the visual block artifacts of JPEC images for Urban Surveillance Systems. The third one is to design a novel filter to recognize vehicle plates (or vehicles) inside the images. Then we improve our filter to identify whether there are objects in these images captured by the cameras, hence the background subtraction is easy to be achieved. At last, our research is to combine the image Hash methods with our proposed filters to retrieve similar duplicate images, which is difficult of accomplishment with only the image Hash methods.

1.2 Thesis Contributions

The main contributions of this thesis can be summarized as five parts, which are described in section 1.2.1 to 1.2.5.

1.2.1 Feature-Based Low-Complexity Intelligent Big Video Data Recording Algorithms for Urban Surveillance Systems

In order to obtain useful information from huge images and videos, many researchers focus on object detection from videos. For example, an application for vehicle search in crowded urban surveillance videos was proposed [5]. Another work proposed a new visual object tracking algorithm using a novel Bayesian Kalman filter with simplified Gaussian mixture [6].

Concerning the video data analytics, researchers focus on compressing every single image frame. For example, some people proposed a method for simultaneously estimating the high-resolution frames and the corresponding motion field from a compressed low-resolution video sequence [7]. Another research hybrids motion-compensation and transform coding schemes, uses Bayesian method to enhance the resolution of compressed video data [8]. In order to cope with the high computing complexity, some researches focus on implementing the computing on hardwired design, such as embedded compression engine targeting the reduction of full HD video transmission bandwidth over the wireless networks [9]; or a CMOS image sensor for tracking the moving objects in region-of-interest and suppressing motion blur [10].

Although there are some work focusing on the single frame compression of video data, currently there is no work done to solve the recording processes of huge video data. We proposed simple and effective methods to dramatically cut the image frame number which needs to be stored up in the hard discs (HD) of the urban surveillance systems. The schemes we proposed include pre-treatment scheme, post-treatment scheme and combined-treatment scheme. To the best of our knowledge, we are the first to propose these schemes.

The novelty of our methods is to only record those important frames. The simulation results show that under normal situations, the occupation space of the HDs could be only a little part of the original one. In this way, the same HDs can record many more days’ information than the existing methods.

A more important aspect of our schemes is the avoidance of complex calculation. The time complexity of our methods is only *O(n)*, which means they are very efficient and economic, hence they are practical to be used in real urban surveillance systems.

1.2.2 Reducing the Visual Block Artifacts of JPEG Images

Since videos are piled up by images, researching on effective image compression techniques is an important research topic. Although the JPEG compression techniques are widely used nowadays, it encounters insurmountable problems. For instance, image encoding deficiencies such as block artifacts have to be removed frequently.

In the mid 80's, the joint collaboration of International Telecommunication Union (ITU) and International organization for Standard (ISO) introduced the standard for the compression of grey scale and colour images. This standard is named the Joint Photographic Experts Group (JPEG) [11]. It has high compression ratio and it is easy to be implemented, hence the JPEG is widely used nowadays [12].

The core codec method of JPEG is Discrete Cosine Transform (DCT). DCT expresses a sequence of data points in terms of a sum of cosine functions oscillating at different frequencies.

The two-dimensional DCT transform is used in JPEG image compression.

In order to get the results of JPEG transform , we programmed in Matlab and simulated the different compression ratio.

In 2000, a new image standard JPEG2000 is emerging. JPEG2000 is the latest image compression standard from the Joint Photographic Expert Group. It was established as an ISO standard in December of 2000 and revised in September of 2004. The image compression standard JPEG2000 is based on the DWT algorithm. JPEG2000 has excellent compression performance while at the same time it provides a rich set of features, e.g., resolution and signal to noise ratio (SNR). In each of the JPEG2000 operational modes, there exists separate wavelet transform [13]. Wavelet analysis is a very important research area in digital image processing. It is a Joint time-frequency analysis and gives both the information in the time and frequency areas.

The process for wavelet transform is to let the image to pass through a low pass filter first, and then pass through a high pass filter. In this way, an image is divided into four parts, which are LL, HL, LH and HH, as shown in Figure 1.1. This is the process of one stage two-dimension wavelet transforms, and the process can be extended further to two stages or more stages.

In this thesis, we have reduced the block artifacts of the urban surveillance systems by proposing a low-complexity and effective method.

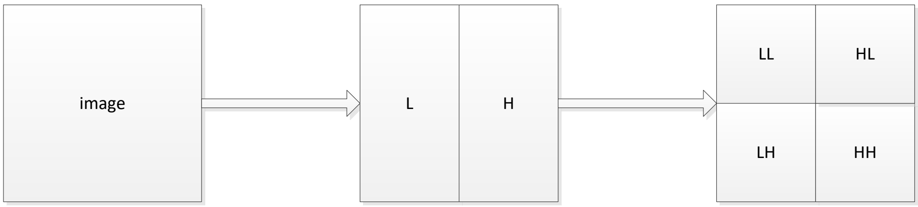

Figure 1.1. One Stage Two-dimension Wavelet transforms

1.2.3 Novel License Plate (or Vehicle) Recognition Methods

Intelligent Vehicle Licence Plate Recognition (VLPR) is an important component of Urban Surveillance Systems. It can be used for automated car identification. Since the vehicle licence plate number is a unique identification for a vehicle, the VLPR is a key technique in most of the traffic related applications, such as road traffic monitoring, security control of restricted areas, automatic parking lots access control, searching for stolen vehicles, etc. The research of VLPR has become a worldwide hot topic.

VLPR algorithms in images are generally composed of three steps. The first step is the license plate localization, which is the extraction of a license plate region from an image. The second step is the segmentation of the plate characters into single character images. The third step is the recognition of each character. Among the three steps, the licence plate recognition (LTR) is usually the most important and difficult module in VLPR systems [14], which influences the accuracy of the system significantly.

Numerous techniques have been proposed for locating the plate through visual image processing [15]. A typical way is vertical edge matching [16]: If the two vertical edges of a licence plate can be detected correctly, four corners of the plate can then be located, so that the license plate can be extracted. Based on the contrast between the grayscale values, [17] also proposed a fast vertical edge based method for LTR, to reduce the computational time of the whole VLPR systems.

A commonly used technology is morphology based license plate detection. A morphology based method is to extract important contrast features as guides to search the desired license plates [18]. In [19], to extract text information from the picture, an adaptive threshold, fractal filter and morphological analysis has been used. Papers [20] and [21] combined edge statistics with morphological steps to eliminate unwanted edges in the images.

Another important technology is colour based methods, which make use of the colour characteristics of the Vehicle Licence Plate. In [22], the image is changed from RGB model into HSV model or HSI model, where the H, S, V, I represent hue, saturation, value intensity respectively. In papers [23] [24], the combination of edge information and plate colour are used to detect the place of vehicle plates.

Some neural network-based vehicle license plate recognition methods are also proposed. These methods attempted to train a classifier to give a proper response to the plate image. In [25], the authors introduced genetic algorithm (GA) to the training process, combined the structure feature with the statistical feature to compensate mutually. In [26], a license plate recognition system based on neural networks was designed and developed. The system used a neural network chip named as CogniMem to recognize license plates. In [27], the authors proposed method based on wavelet transform to decompose the images into different layers, and then use the low frequency images to combine with neural networks. In this way, the recognition time of the license recognition system can be reduced.

Another method was proposed based on the horizontal and vertical frequency energy differences. In [28], the authors used Daubechies wavelet to calculate the vertical and horizontal frequency energy curves, since the frequency is the highest near the number plate.

Our research has investigated the frequency characteristics of vehicle images, in order to design a novel filter to search the images, and to filter out the vehicle license plate area or vehicles area. Since our research focused on the frequency characteristics of these images, it made the search process comparably simple when comparing to current methods in the literature.

1.2.4 Low-Complexity Automatic Background Subtraction Techniques

Due to the tremendous data generated by urban surveillance systems, the demand for big data oriented low-complexity automatic background subtraction techniques is increasing.

Background subtraction method firstly builds the background model, and then uses the current frame image to subtract the background model. In this way the targets are picked out.

The background subtraction method depends mainly on the background modelling techniques, in which Gaussian mixture models (GMMs) are now the popular methods. It depends on the moving situation of objects.

We propose an automatic background subtraction method which is not depend on the moving situation of objects. In this way the computer can automatically renew the image as the new background when no object is detected. This method is simple and also robust to the impact of light changing.

1.2.5 Fast Saliency-Aware Image Hash for Near Duplicate Images Retrieval

Hash methods have been widely used in image search. In image Hash methods, unique codes which are extracted from images are used as the identifications for images.

The basic principle of image Hash methods determine that the positions of the main objects inside an image should be unmoved, which limits the wide usage of image Hash.

In this part, we will combine image Hash method with our proposed method to find near duplicate images. In this way, the slight moving of the object will be detected and near duplicated images with same source will be determined automatically. By using this simple algorithm, our proposed Image Hash method is fast, saliency-aware and practical.

1.3 List of Publications

- L. Hu, Q. Ni, F. Yuan, “Big data oriented novel background subtraction algorithm for urban surveillance systems”, Big Data Mining and Analytics, Volume: 1, Issue: 2, Pages: 137-145, June 2018.

- L. Hu, Q. Ni, “IoT-Driven Automated Object Detection Algorithm for Urban Surveillance Systems in Smart Cities”, IEEE Internet of Things Journal, Volume: 5, Issue: 2, Pages: 747-754, April 2018.

- L. Hu, Q. Ni, “Low-Complexity Big Video Data Recording Algorithms for Urban Surveillance Systems”, International Journal of Data Mining & Knowledge Management Process, Vol. 6, Number 6, pp. 1-16, November 2016.

- L. Hu, Q. Ni, “Big Data-Driven Fast Reducing the Visual Block Artifacts of DCT Compressed Images for Urban Surveillance Systems”, International Journal of Data Mining & Knowledge Management Process, Vol. 6, Number 4, pp. 19-29, July 2016.

CHAPTER 2

LOW-COMPLEXITY BIG VIDEO DATA RECORDING ALGORITHMS FOR URBAN SURVEILLANCE SYSTEMS

Part of this chapter was published as a research paper below:

L. Hu, Q. Ni, "Low-Complexity Big Video Data Recording Algorithms for Urban Surveillance Systems", International Journal of Data Mining & Knowledge Management Process, Vol. 6, Number 6, pp. 1-16, November 2016.

Big Video data analytics and processing are becoming increasingly important research areas because of infinite generation of massive video data volumes all over the world. In this chapter, by utilizing Bayesian-based importance analysis, we propose a set of novel, simple but effective video recording methodologies and intelligent algorithms to solve the so-called big video data volume problem in urban surveillance systems. The complexity of our proposed algorithms are only O(n), hence they can be easily implemented in real video urban surveillance systems without complicated computational cost. The simulation results show amazing recording efficiency. Our methods can dramatically reduce the hard discs (HD) occupation space requirement so that the recording time of HDs can be greatly enlarged.

2.1 Introduction

Big data is used to describe a massive volume of both structured and unstructured data sets which is very large and beyond the ability of traditional database tools to capture, manage, record and process them. According to the 3V's definition, big data has three main characteristics, which are Volume, Velocity and Variety [1][29]. Volume refers to an extreme amount of data; the examples of big data might be petabytes (1,024 terabytes) and exabytes (1,024 petabytes) of data consisting of billions to trillions of records, from different sources (e.g. Web, sales, customer contact centres, social media, mobile data and so on). Velocity describes how fast the data is produced and must be processed. Variety means a wide range of data types. For example, data can be collected/stored in multiple formats, which may include image, audio, video, texts and other forms of data.

Among all these different varieties of big data, videos are the biggest. According to Cisco's statistics, images and videos make up about 90 percent of overall IP traffic

[30]. Hence video data analysis receives an increasing attention due to its fast-growing applications and huge data volumes. For instance, the data volume of all video surveillance devices in Shanghai, China is up to 1 TB every day [31] and is expected to keep growing substantially in the future. The explosive increasing number of video resources has brought an urgent need to develop intelligent methods to reduce the pressure on the database management processes [32].

In order to dig out useful information from huge image and video data, many researchers focus on object detection from videos. For example, an application for vehicle search in crowded urban surveillance videos was proposed [33]. Another work proposed a new visual object tracking algorithm using a Bayesian Kalman filter with simplified Gaussian mixture [34].

Concerning the video data analysis, most researchers in the literature focus on compressing every single image frame. For example, some people proposed a method for simultaneously estimating the high-resolution frames and the corresponding motion field from a compressed low-resolution video sequence [35]. Another research combines motion-compensation and transform coding schemes, and uses Bayesian method to enhance the resolution of compressed video data [36]. In order to cope with the high computational complexity, some researches focus on implementing the computation on a hardwired design, such as embedded compression engine targeting the reduction of full high definition video transmission bandwidth over the wireless networks [37]; or a CMOS image sensor for tracking the moving objects in region-of-interest and suppressing motion blur [38].

Although there is much work focusing on the detecting information from frames and videos, to the best of our knowledge, currently there is little work done to reduce data volume in the video recording processes. Even though it is important to deal with the single images, efforts need to be done on the recording processes to solve the big video data recording problem. Nowadays the big video data increases rapidly, which makes its volume much higher than the potential hard discs (HD) capacity. In this chapter, we propose a set of simple and effective methods to intelligently and dramatically reduce the video frame number which are needed to be stored up in the HDs for the urban surveillance systems. To the best of our knowledge, we are the first to propose these schemes.

2.2 Novelty and Contributions

In this chapter, we propose several new simple but effective methods to dramatically reduce the space of the HDs in order to record the huge amount of video data generated by urban surveillance systems.

Different from the current efforts in the literature which focus on single video frame compression, the novelty of our methods is to analyse and to only record those important frames according to feature requirement. Using Bayesian-based importance analysis, we propose three intelligent mechanisms, called pre-treatment scheme, post-treatment scheme and combined-treatment scheme. We first determine which are unimportant frames and then remove part of them in the recording process to dramatically reduce the video recording data volume while at the same time to keep acceptable visual quality. The simulation results demonstrate that using our proposed methods, the occupation space of the HDs can be significantly reduced.

An important advantage of our schemes is their low complexity features. The complexity of our methods is only O(n). This means that they are very efficient, and hence they are easy to be adopted and deployed in real video urban surveillance systems.

2.3 Urban Surveillance System Model

Urban surveillance systems are widely used now to improve the intelligence of cities. The normal applications include counter surveillance in banks, entrance monitoring of safeguard regions, traffic monitoring of crossroads, and so on. In most cases, the cameras are fixed and the surveillance regions are also located constantly. A typical diagram of urban surveillance system is shown in Figure 2.1. In current urban video surveillance systems, there are huge number of cameras working 24 hours per day, and 7 days a week. Every camera focuses on a fixed area, in order to show the scenarios instantly, clearly and fluently. The frame per second (FPS) of every camera is normally kept at 30. In order to get higher quality for instant monitoring, some cameras increase the frame rate from 30 FPS to 60 FPS, or even higher (some cameras may even adopt 100 FPS). In these systems, digital video recorders (DVRs) are used to record the video frames; the most common number of inputs for a DVR is 1, 2, 4, 8, 16 and 32. Obviously the key components of the DVRs are huge HDs. Although various compression techniques are currently used to reduce the size of every frame, the demand for high capacity HDs still increases rapidly due to the huge number of frames generated and to be generated explosively in the future.

A widely used way of urban surveillance systems is to monitor the traffic accident scenes of the roads. Research work has shown that there are four factors to influence safety management of traffic accident scenes, which are human factor, vehicle factor, road and environment factor, and management factor. The human factor, vehicle factor and management factor normally do not change between a 24 hour period; but the roads are more crowded at the peak time periods, it can worsen the road and environment factor and affect the psychological diathesis of drivers, which makes the probability of traffic accidents occurring to increase [39].

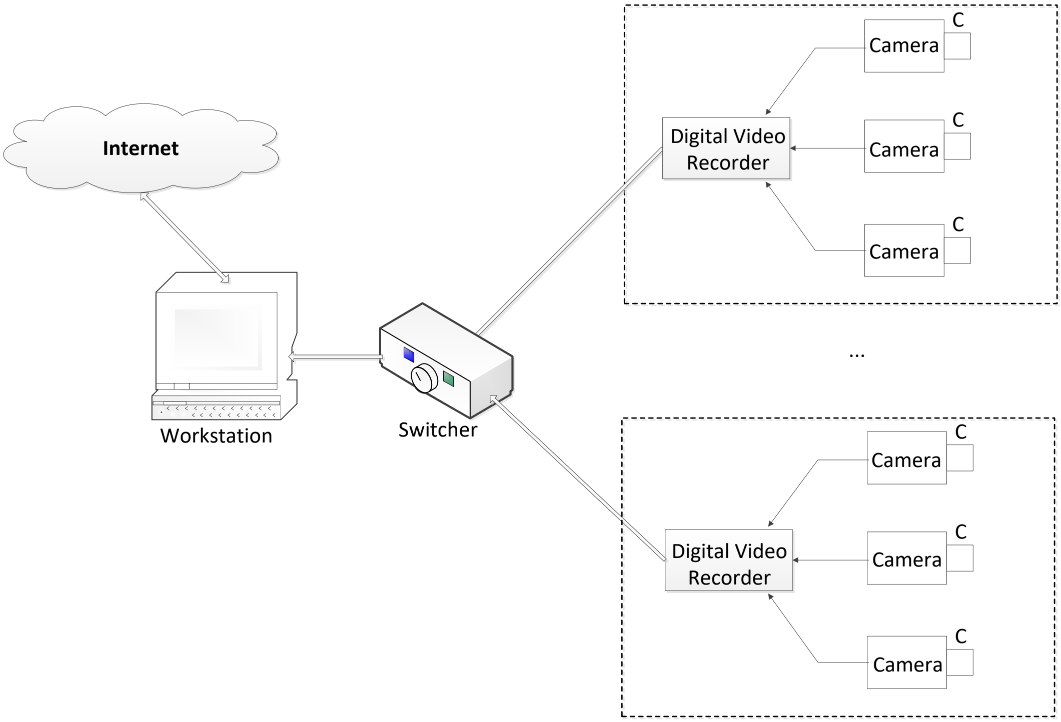

Figure 2.1. The Basic Diagram of Urban Surveillance Systems

2.4 Our Proposed Schemes

2.4.1 Bayesian-Based Analysis

Concerning the urban video surveillance systems, in order to reduce the number of recorded video frames, we propose the idea of only recording those frames which are potentially important. But, how do the systems know or determine which frames are important without extremely complex analysis of huge video frames data generated and to be generated continuously? First, we use Bayesian Theory to provide a theoretical guidance on how to design simple approaches of importance analysis. Let us assume that there is a camera installed at a bus station to monitor the traffic situations. We denote A_1 and A_2 as the events if the time duration is a peak time period and a non-peak time period, respectively. B is defined as the event if an accident happens.

A_1: The traffic is crowded (i.e. peak time).

A_2: The traffic is not crowded (i.e. non-peak time).

B: The event when an accident occurs.

Without loss of generality, we assume that the probabilities of peak time and non-peak time are the same, that is $P(A_1)=0.5$ and $P(A_2)=0.5$. Since evidences show that it is more prone for an accident to appear when the traffic is crowded [11], we can assume the probability of an accident happening at peak time is $P(B|A_1)=0.03$ and the probability of an accident happening at non-peak time is $P(B|A_2)=0.01$. As defined, the probability of an accident happening is P(B). Now the problems are, at what time periods do an accident happen mostly? What is the probability that it happens in the peak time period? What is the probability that it happens in non-peak time period?

According to Bayesian rule, the probability of the accident happening in peak time period can be calculated as:

$$P(A_1|B) = \frac{P(A_1)P(B|A_1)}{P(A_1)P(B|A_1)+P(A_2)P(B|A_{\ 2})} = \frac{0.5\times 0.03}{0.5\times 0.03+0.5\times 0.01} = 75\% \tag{1}$$

The probability of an accident happening in non-peak time period is obtained as:

$$P(A_2|B) = \frac{P(A_2)P(B|A_2)}{P(A_1)P(B|A_1)+P(A_2)P(B|A_2)} = \frac{0.5\times 0.01}{0.5\times 0.03+0.5\times 0.01} = 25\% \tag{2}$$

The Bayesian rule is guidance for us to cut some unimportant frames in the recoding processes. From the above example we conclude that the probability of an accident happening in peak time period is much higher than that in off-peak time period even when the probabilities of peak time and off-peak time are the same. Although the high quality frames can be shown at 30 FPS or even higher instantly for the surveillants to keep watching on the scenarios in real time, it is not necessary for all the huge number of video frames to be recorded into the HDs.

2.4.2 Pre-Treatment Scheme

Our pre-treatment scheme is to divide a whole day into two parts: peak time period and non-peak time period. How to divide the time period is dependent on different scenarios. In the peak time period, we keep all the frames to be recorded into the HDs, which is normally 30 FPS. In this way, the comparably important frames are kept without loss, because they have higher probability to be reviewed in the future. On the other hand, we may cut some frames in the non-peak time periods, that is, to reduce the FPS. Even the FPS is as low as 1, it will effectively record the scenarios with human being's activities. Considering 1 FPS is only around 3% of the normally 30 FPS, the frames to be recorded could be reduced dramatically. The pseudo-code of our proposed pre-treatment is outlined in Algorithm 1.

Algorithm 1: Pre-treatment algorithm

Input: The video frames generated by cameras

Return: The video frames needed to be recorded to HD

```
Check current time t
if t belongs to peak hours
```

```
3.  Goto 6
4.  Else
5.  Cut the FPS to the pre-defined number of non-peak time period
6.  End
7.  Record the video frames to HD
```

As we already know, a typical camera generates 5.4G bytes video data per day. Hence, with a commonly 25 inputs DVR, it will need 4050G bytes space to record the videos only for 30 days, which is a tremendous volume. In order to save the space of HDs, currently the DVRs have the program for the surveillants to choose which days' video should be recorded. This is obviously a simple but coarse method, which may cause some important frames to get lost if we cut frames randomly.

Using our scheme, the calculation could be low-cost. Our model is simple to be implemented within the video processor, concerning the significant efficiency of cutting big video data. With the same HD space, obviously the total recording time will be extended.

2.4.3 Post-Treatment Scheme

In some cases, it is not practical to divide a day into two parts, as every time instant along the whole day could be equally important. Hence we propose a new post-treatment scheme. In most circumstances, people need to review the recorded video shortly after something happened. In practice, the probability of the need to review the recorded videos in the near several days is the highest. When the time flies away, the need to review the history videos reduces rapidly. By taking this into account, we propose a post-treatment scheme where our idea is to keep all the generated frames recorded into the HDs initially, which is normally 30 FPS. Since the importance of the recorded video frames reduces rapidly with time passing, we thus introduce a cut-off day. For instance, if we define the cut-off day as 3 days, this means that after the 3 days we could discard some of the recorded video frames, from normal 30 FPS to a lower FPS. In this way, the re-recorded files will then be significantly smaller to be saved concerning the space of the HDs.

The process of our proposed post-treatment is depicted in Algorithm 2.

Algorithm 2: Post-treatment algorithm

Input: The recorded video frames

Return: The video frames needed to be re-recorded to HD

```
Check current day d after the frames recorded day
if d = cutoff day
    Discard some frames according to pre-defined FPS
    Re-record the video frames to HD
Else
End
```

2.4.4 Combined-Treatment Scheme

Finally, our pre-treatment and post-treatment may be combined together to achieve better results. The process of the combined-treatment scheme is shown in Figure 2.2.

When an HD receives the video frames occurred from the surveillance cameras, it can begin to run the pre-treatment scheme, that is, to cut some frames from the non-peak hours and save the resulted data into the HD to generate the current recording files.

At later days, the system will check the recorded files to see whether or not the cut-off day is due. When the cut-off day is due, the post-treatment scheme will begin to work. The recorded files may be re-recorded with the pre-defined lower FPS; hence the size of the files will be reduced again.

By combining intelligent video analysis with high performance computing technology, our proposed methods will help to develop intelligent surveillance video systems and result in huge economic benefit.

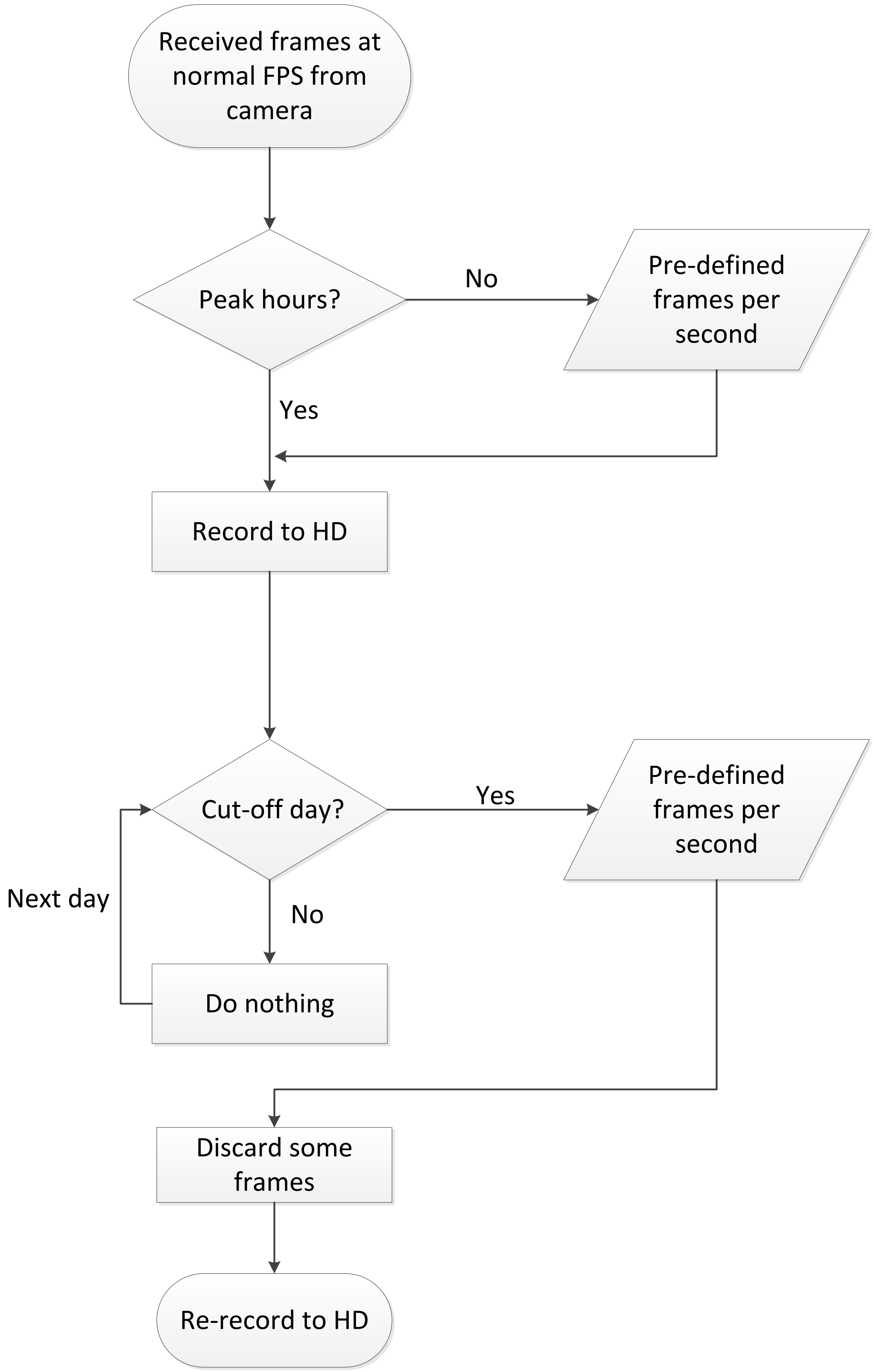
Received frames at normal FPS from camera
Peak hours?
No
Pre-defined frames per second
Yes
Record to HD
Cut-off day?
Yes
Pre-defined frames per second
Next day
No
Do nothing
Discard some frames
Re-record to HD

Figure 2.2. The Flowchart of the Combined-Treatment

2.4.5 Complexity Analysis

Let n denote the number of video frames. As for the time complexity of the pre-treatment scheme, the calculation time cost is only linear with the off-peak time's number of video frames, hence the time complexity of this scheme is only O(n).

As for the post-treatment scheme, the calculation time cost is linear with the pre-defined re-record frames' number after cut-off day, hence the time complexity of this scheme is also O(n).

For our combined-treatment method, the calculation time cost is also linear with the frame number at off-peak time or after the cut-off day, hence the time complexity of our combined-treatment scheme is still O(n).

2.5 Simulation Results

2.5.1 Pre-Treatment Scheme

We simulate our schemes and plot the simulation results in figures. In order to reflect the most common scenarios, we set the FPS at peak time period as 30. The FPS at off-peak time period may be set from 1 to 30. The peak time hours per day could be chosen from 0 to 24. The simulation results are illustrated in 3-dimensional plots. They are shown in Figure 2.3 and Figure 2.4.

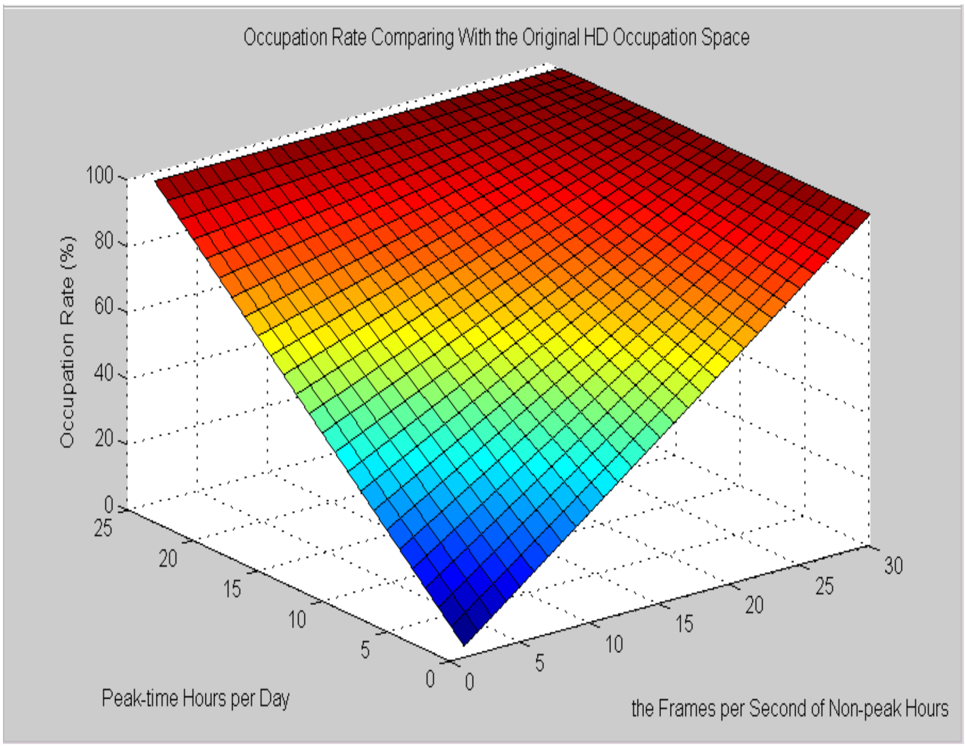

Figure 2.3. Occupation Rate Comparing with the Original HD Occupation Space (3D)

Figure 2.3 illustrates the occupation rate comparing with the original HD occupation space for a 24 hours period. X-axis is the frames per second of non-peak hours; Y-axis shows the peak time hours per day; and Z-axis shows the results of the HD occupation rate in percentage comparing with the original HD occupation space.

Figure 2.4 shows the gains in recording time comparing with the original time period with the same HD. X-axis is the frames per second of non-peak hours, which ranges from 1 to 30. Y-axis shows the peak time hours per day, which is from 0 to 24. Z-axis shows the gains in recording time with the same HD, which are displayed in percentage.

Figure 2.3 and Figure 2.4 show significant improvement of the urban video surveillance systems. When we choose less peak-time per day and fewer frames per second of non-peak time period, the occupation rate comparing with the original HD occupation space falls down sharply. At the same time, the gains in recording time rise up at very high speed, concerning the same HD space. While our scheme is very simple and easy to be implemented, the performance gain is amazing.

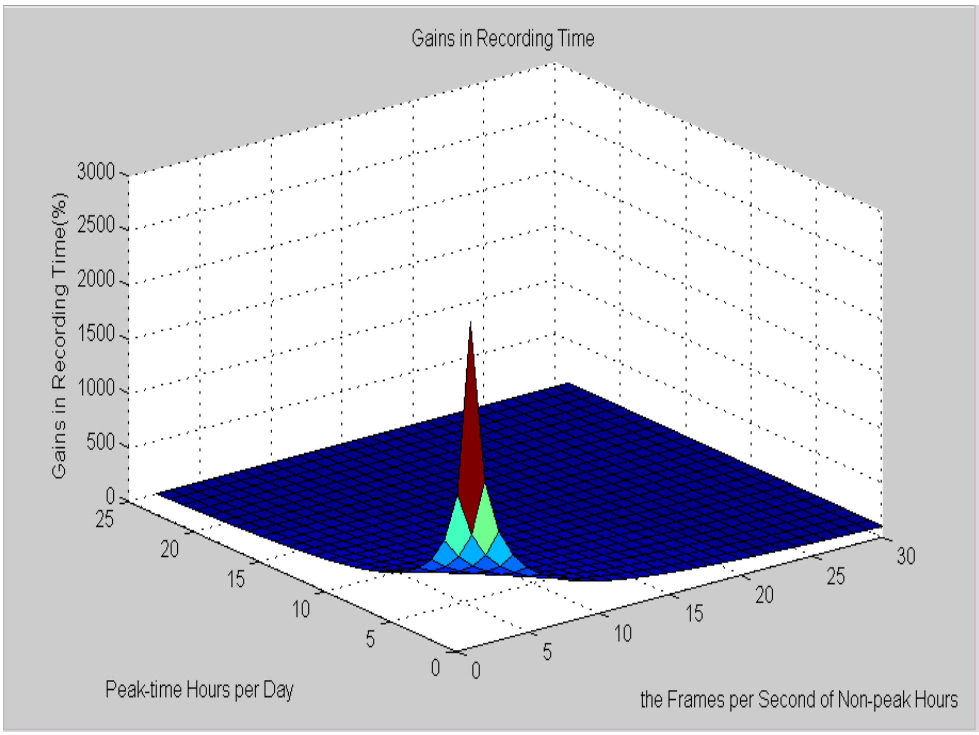

Figure 2.4. Gain of Recording Time (3D)

Since Figures 2.3 and Figure 2.4 are 3-dimensional figures, it is not easy for people to investigate further details. We choose the situations of peak time from 0 to 8 hours per day, which may cover most common scenarios, to plot the results in 2-dimensions. The 2-dimensional results are plotted in Figures 2.5 and Figure 2.6.

Figure 2.5 is the occupation rate comparing with the original HD occupation space for a 24 hours period. X-axis is the frames per second of non-peak hours, and Y-axis shows the result of the HD occupation rate in percentage comparing with the original

HD occupation space. For different peak-time hours per day, the results are plotted in different lines.

Figure 2.6 illustrates the recording time gain comparing with the original time period with the same HD. X-axis is the frames per second of non-peak hours, and Y-axis shows the gains in recording time in percentage. For different peak-time hours per day, the results are plotted in different lines.

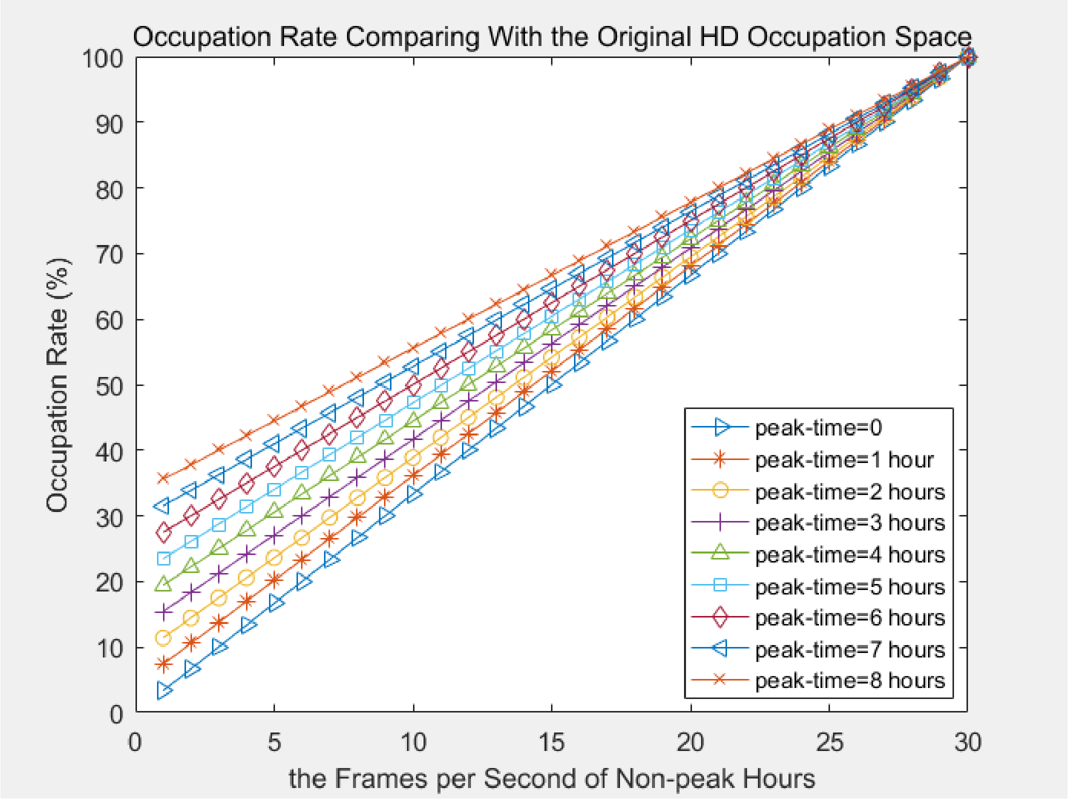

Figure 2.5. Occupation Rate Comparing with the Original HD Occupation Space

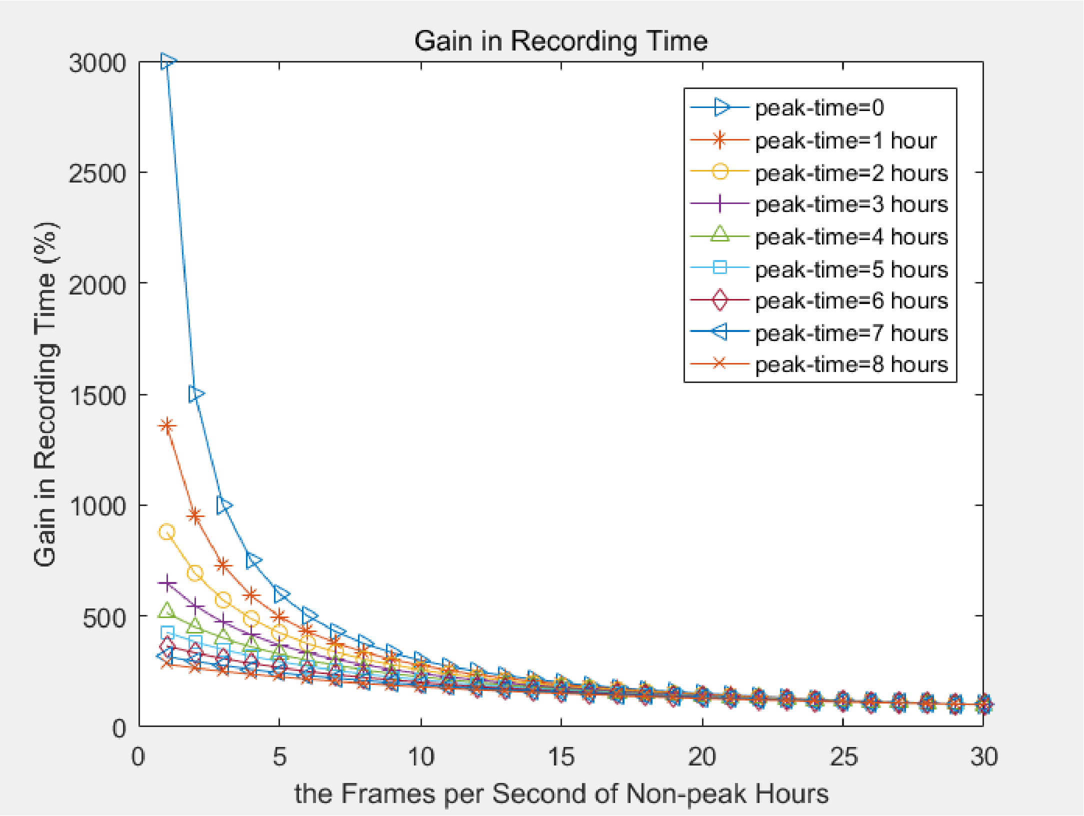

Figure 2.6. Gains of Recording Time

As shown in Figures 2.5 and Figure 2.6, at each different peak time hours per day, the occupation rate comparing with the original HD occupation space is a linear result, the lower FPS of non-peak time hours leads to the lower occupation rate.

At the same time, the time gain comparing with the original time period shows the amazing results. For example, when a camera watches at a never crowded road, the peak time may be set at 0. When the recorded FPS is set to 1, the system will not lose the main activities of cars and pedestrians. The occupation rate may be as low as around 3.3% of original HD space for a 24 hours period; and the recording time gain may be around 30 times compared to the original time with the same HD space.

2.5.2 Post-Treatment Scheme

Again, we plot here the simulation results. We still set the original FPS as 30. The cut-off days could be chosen from 1 to 30. The FPSs after the cut-off days are set from 1 to 30. The calculation is based on the original recorded period of 30 days. The simulation results are also plotted in 3-dimensions. They are illustrated in Figure 2.7 and Figure 2.8.

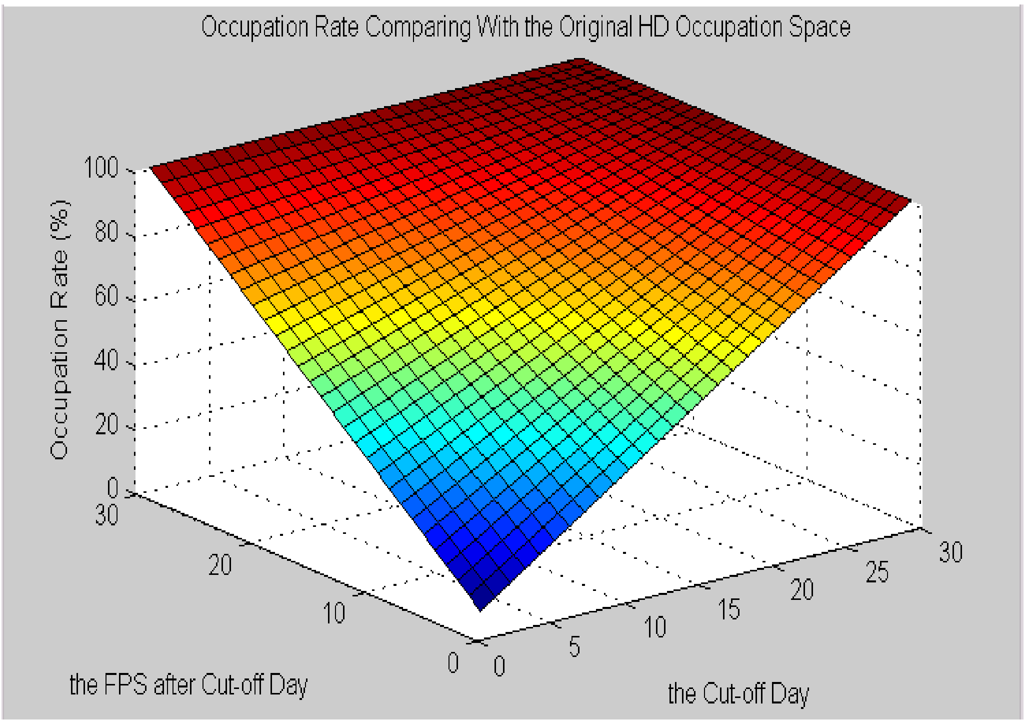

Figure 2.7. Occupation Rate Comparing with the Original HD Occupation Space (3D)

Figure 2.7 is the occupation rate comparing with the original HD occupation space for a 30 days period. X-axis shows the pre-defined cut-off days. After the cut-off day, the recorded video frames will be discarded partly. Y-axis shows the re-recorded FPSs after the cut-off day and Z-axis are the results of the HD occupation rate in percentage comparing with the original HD occupation space.

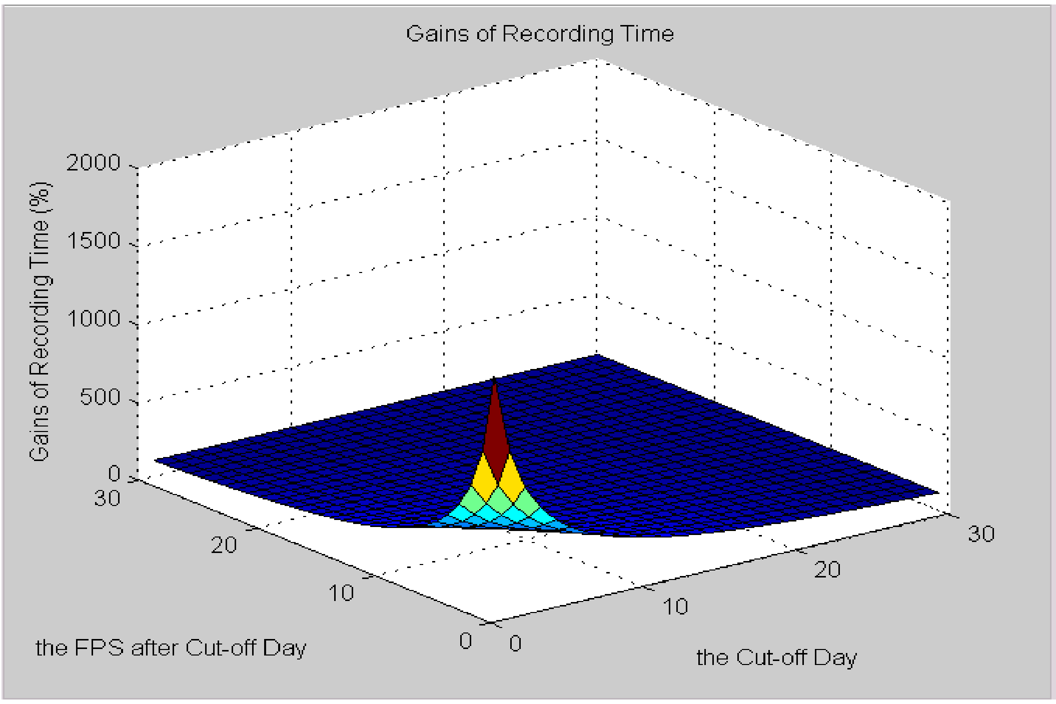

Figure 2.8. Gains of Recording Time(3D)

Figure 2.8 shows the gain of recording time in percentage using our post-treatment method, concerning using the same HD occupation space of the original period of 30 days. X-axis are the cut-off days. Y-axis shows the re-recorded FPSs after the cut-off day. Z-axis shows the gain of recording time in percentage.

Figures 2.7 and Figure 2.8 also show significant improvement of the urban video surveillance systems. The occupation rate comparing with the original HD occupation space falls down sharply when we choose smaller cut-off day and smaller re-recorded FPS after the cut-off day. At the same time, the gain in recording time rises up sharply in comparison with the same HD space of original 30 days' images.

Again we choose the cut-off days from 1 day to 8 days to plot the 2-dimensional simulation results as shown in Figures 2.9 and Figure 2.10.

Figure 2.9 shows the occupation rate comparing with the original HD occupation space for a 30 days period using our post-treatment method. X-axis are the re-recorded FPSs after a cut-off day, and Y-axis shows the results of the HD occupation rate in percentage comparing with the original HD occupation space. For different cases of cut-off day settings, the results are plotted in different lines.

Figure 2.10 illustrates the gain of recording time, concerning using the same HD space of the original period of 30 days. Again X-axis shows the re-recorded FPSs after the cut-off day and Y-axis are the results.

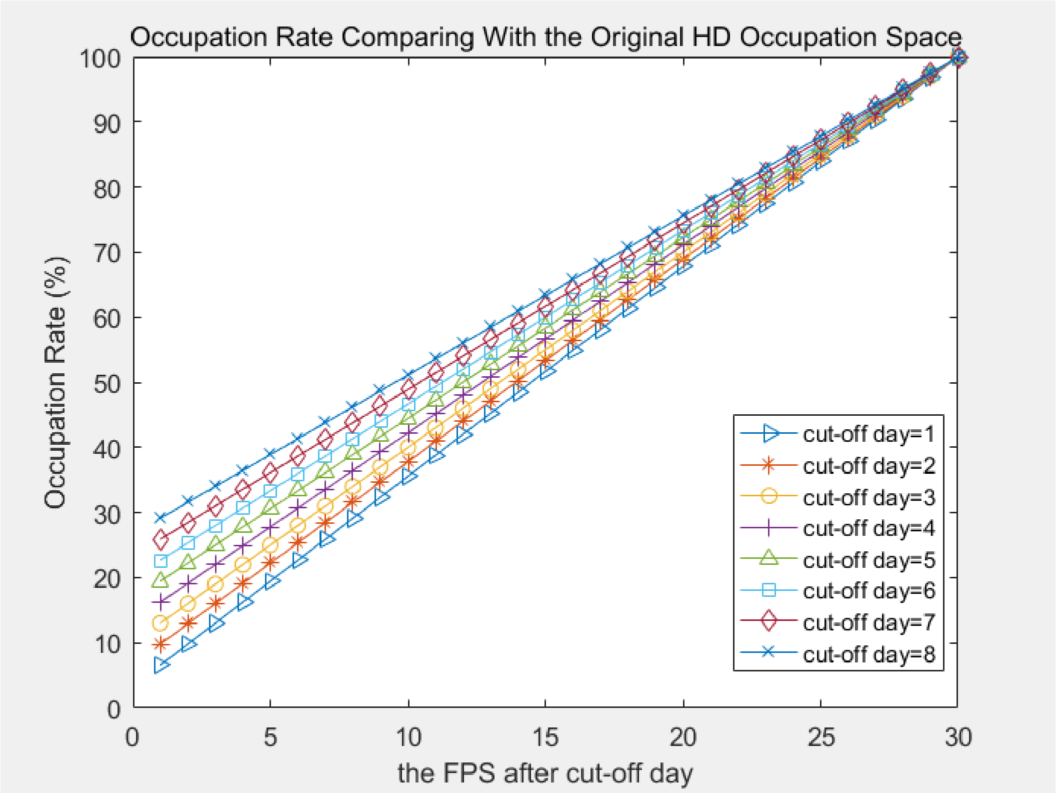

Figure 2.9. Occupation Rate Comparing with the Original HD Occupation Space

As shown in Figures 2.9 and Figure 2.10, at each different cut-off day, the occupation rate comparing with the original HD occupation space is a linear result. The gains of recording time again show amazing results. For instance, when the recorded FPS after cut-off day is set at 1, it will not lose the main activities of human being and cars. In these cases, the gain of recording time in percentage are extremely high, which may easily reach several hundred, or even higher than one thousand percentage in common use scenarios.

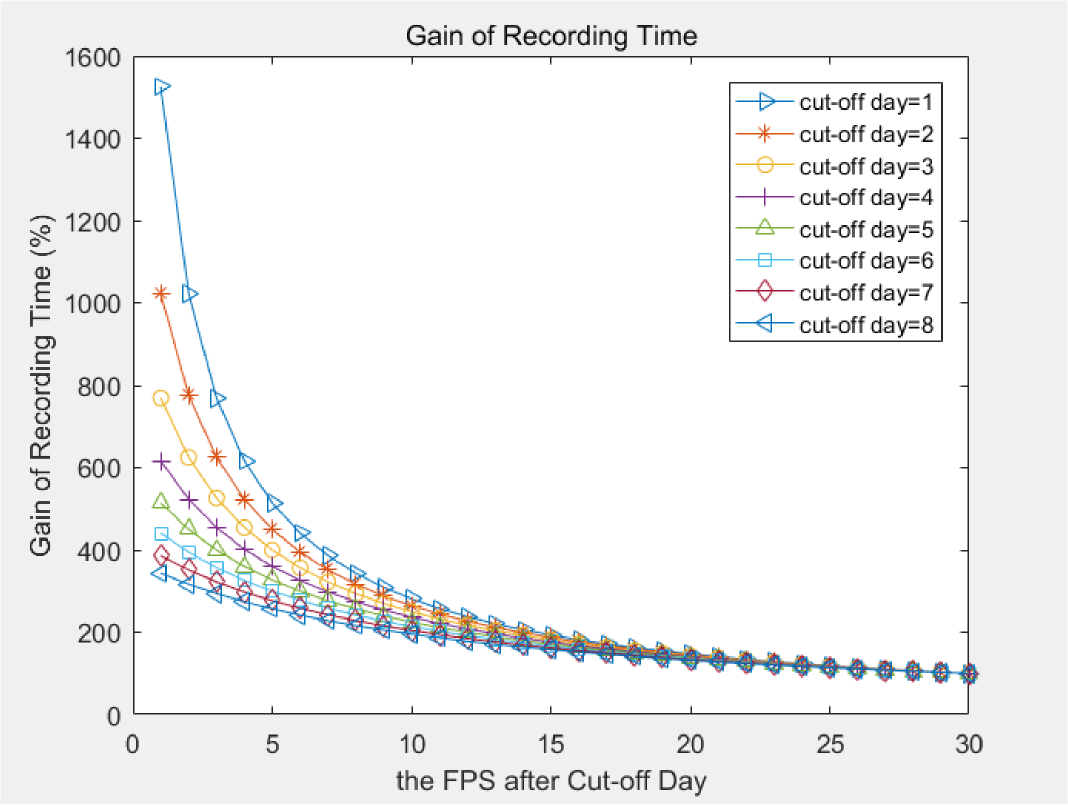

Figure 2.10. Gains of Recording Time

2.5.3 Combined-Treatment Scheme

For our combined-treatment scheme, in order to get the occupation rate comparing with the original HD occupation space for a 30 days period, we may use X-axis to represent the peak-time hours per day, and use the Y-axis to show the cut-off day. At peak time hours and before the cut-off day, the original FPS is still set at 30. At off-peak time or after the cut-off day, the FPSs are all set to 1. Again, the Z-axis shows the occupation rate comparing with the original HD occupation space for a 30 days period. The 3-dimensional simulation results are shown in Figure 2.11.

Figure 2.12 shows the gain of recording time using our combined-treatment method, as compared to concerning using the same HD occupation space of the original period of 30 days. The X-axis represents the peak-time hours per day, and the Y-axis shows the cut-off days. The Z-axis shows the results in percentage, which are the gains of recording time comparing with using the same HD space of the original video frames of 30 days.

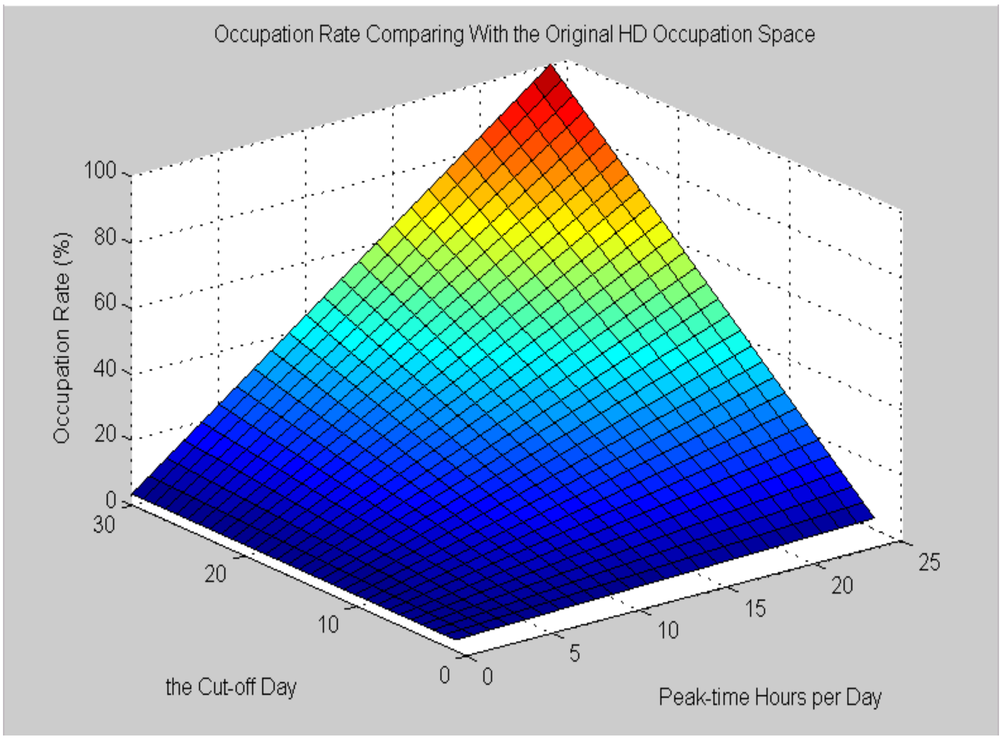

Figure 2.11. Occupation Rate Comparing with the Original HD Occupation Space (3D)

Figures 2.11 and Figure 2.12 show significant improvement of the urban video surveillance systems. The occupation rate comparing with the original HD occupation space falls down quickly when we choose smaller peak-time hours per day and smaller cut-off days, and the corresponding gains of recording time again increase sharply. The figures show that the proposed combined-treatment scheme achieves better efficiency than the pre-treatment method or post-treatment method.

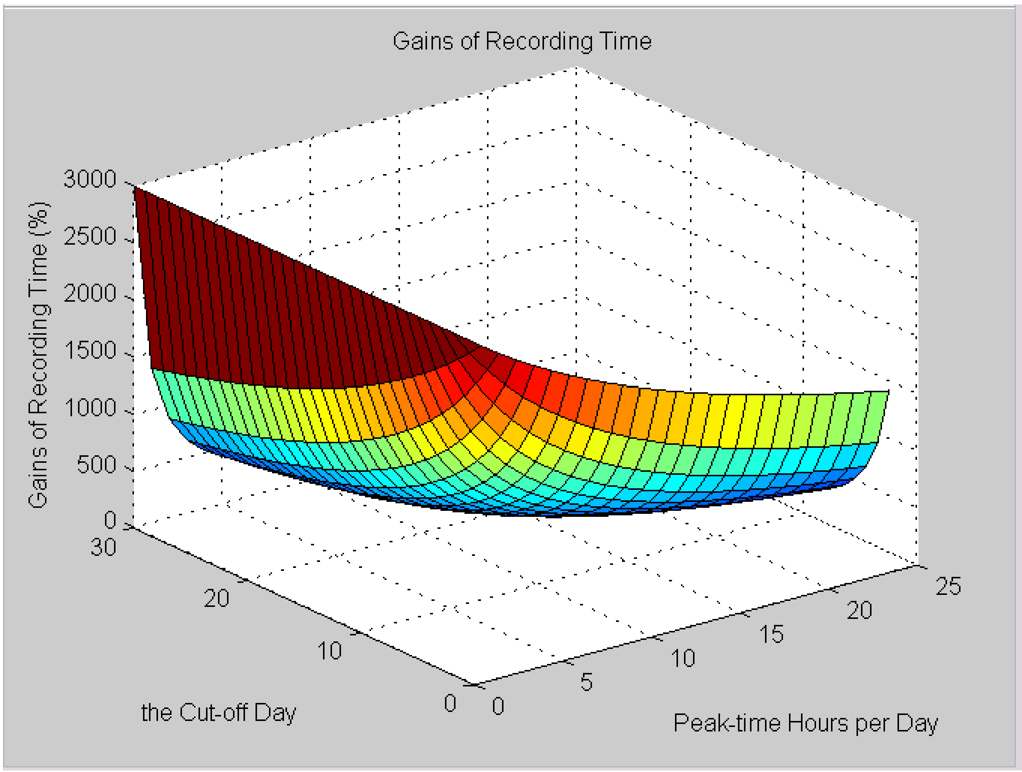

Figure 2.12. Gains of Recording Time (3D)

Now we choose the cut-off day from 1 to 8 days situations, which may cover most common use scenarios, to plot the results in 2-dimensions. The results are shown in Figure 2.13 and Figure 2.14.

Figure 2.13 shows the occupation rate comparison by varying peak hours per day and cut-off day duration using our combined treatment method. During the peak time hours and before the cut-off day, the original FPS is set at 30. During the off-peak time or after the cut-off day, the FPSs are all set at 1 FPS. X-axis is the peak-time hours per day, and Y-axis is the results of the HD occupation rate in percentage comparing with the original HD occupation space. For different settings of cut-off days, the results are plotted in different lines.

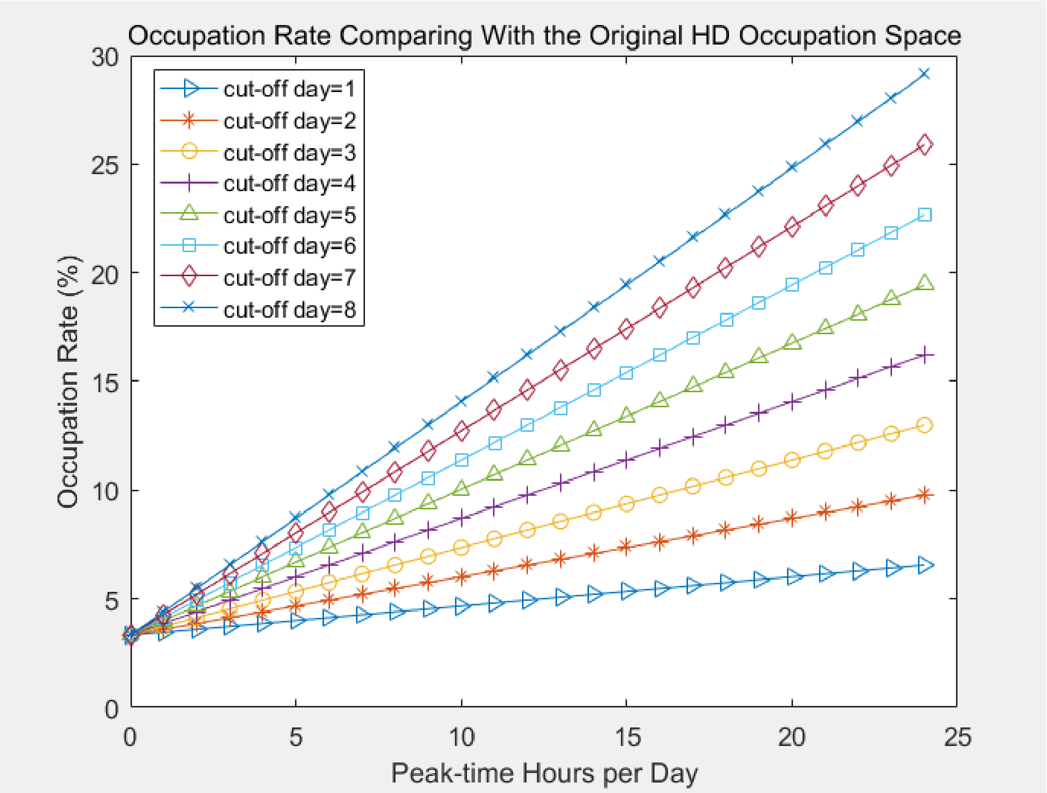

Figure 2.13. Occupation Rate Comparing with the Original HD Occupation Space

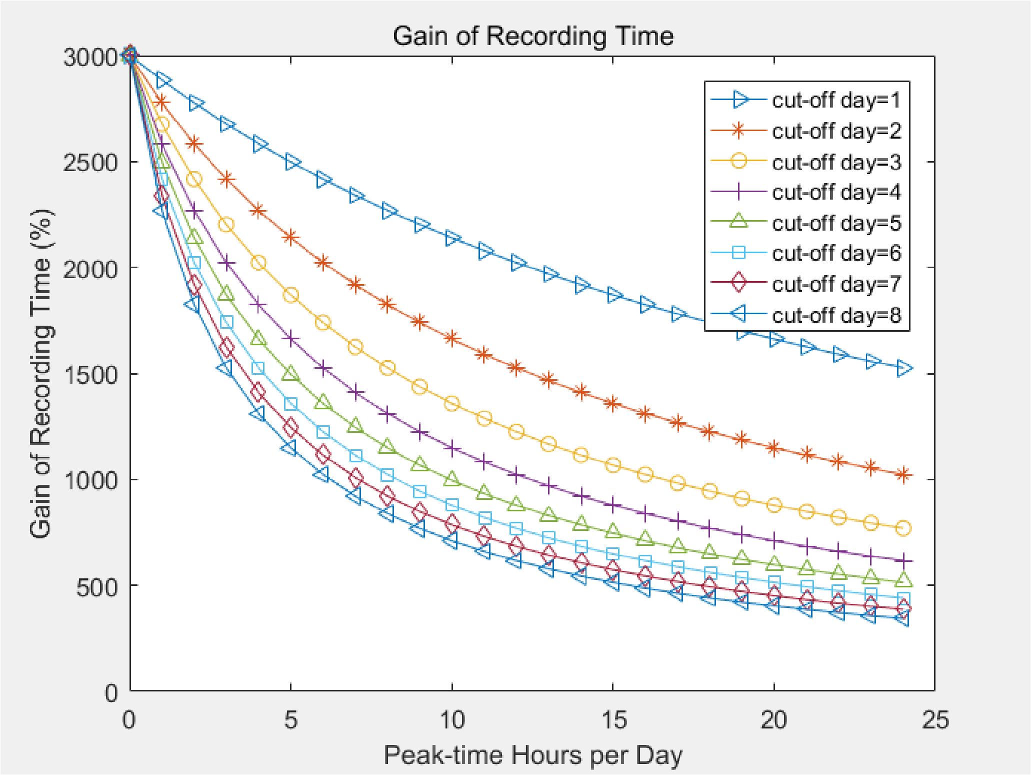

Figure 2.14. Gain of Recording Time

Figure 2.14 illustrates the gain of recording time between the proposed combined-treatment scheme and the existing method. Again during the peak time hours and before the cut-off day, the original FPS is set to 30. During the off-peak time or after the cut-off day, the FPSs are set to 1. X-axis is the peak-time hours per day, and Y-axis shows the results, which are the gain of recording time in percentage. For different sets of cut-off days, the results are plotted in different lines. The gains of recording time in percentage reach several thousand. This indicates that our scheme can help to save the HD storage space dramatically.

2.6 Summary

In this chapter we proposed several simple yet efficient video data recoding algorithms to be used in urban surveillance systems. The main idea of our schemes is to intelligently record the important video frames with the number of unimportant video frames to be reduced as small as possible. Our proposed methods are very easy to be implemented on real time urban surveillance systems with low complexity and can efficiently save huge amount of the HDs storage space.

The simulation results demonstrated that, using our combined-treatment scheme, the occupation rate may be as low as around 3% of original HD space. The gains of recording time may easily reach several thousand percentages, comparing with the same HD space recording the original video frames of 30 days period.

Since currently the urban surveillance systems sometimes only record key frames to reduce the space of HDs, our method can combined with the key frames techniques to further reduce the frames which are needed to be recorded.

CHAPTER 3

BIG DATA-DRIVEN FAST REDUCING THE VISUAL BLOCK ARTIFACTS OF DCT COMPRESSED IMAGES FOR URBAN SURVEILLANCE SYSTEMS

Part of this chapter was published as a research paper below:

L. Hu, Q. Ni, "Big Data-Driven Fast Reducing the Visual Block Artifacts of DCT Compressed Images for Urban Surveillance Systems", International Journal of Data Mining & Knowledge Management Process, Vol. 6, Number 4, pp. 19-29, July 2016.

The Urban Surveillance Systems generate huge amount of video and image data and impose high pressure onto the recording disks. It is obvious that the research of video is a key point of big data research areas. Since videos are composed of images, the degree and efficiency of image compression are of great importance. Although the DCT based JPEG standard are widely used, it encounters insurmountable problems. For instance, image encoding deficiencies such as block artifacts have to be removed frequently. In this chapter, we propose a new, simple but effective method to fast reduce the visual block artifacts of DCT compressed images for urban surveillance systems. The simulation results demonstrate that our proposed method achieves better quality than widely used filters while consuming much less computer CPU resources.

3.1 JPEG Introduction

The widespread use of video cameras in Urban Surveillance Systems has resulted in an enormous amount of image and video data, hence the compression of those image and video data become more and more important in big data research areas. In the mid 80's, the joint collaboration of International Telecommunication Union (ITU) and International organization for Standard (ISO) introduced the standard to compress images which is called the Joint Photographic Experts Group (JPEG) [40]. The core codec method of JPEG is Discrete Cosine Transform (DCT). It may get high compression ratio and it is easy to be implemented, hence the DCT based JPEG is widely used [41]. DCT is a Fourier-related transform expresses, it sum up a sequence of cosine functions at different frequencies.

The two-dimension DCT transform is:

$$F(k,l) = a_k a_l \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m,n) \cos\frac{(2m+1)k\pi}{2M} \ \cos\frac{(2n+1)l\pi}{2N} \tag{3.1}$$

$$k = 0,1,2 \ldots M-1; \qquad l = 0,1,2 \ldots N-1$$

$$a_k = \begin{cases} \frac{1}{\sqrt{M}} \ , & k = 0 \\ \sqrt{\frac{2}{M}} \ , & 1 \le k \le M-1 \end{cases} ; \qquad a_l = \begin{cases} \frac{1}{\sqrt{N}} \ , & l = 0 \\ \sqrt{\frac{2}{N}} \ , & 1 \le l \le N-1 \end{cases}$$

The two-dimension DCT transform is used in JPEG image compression. And the block diagram of DCT based JPEG transform can be shown below:

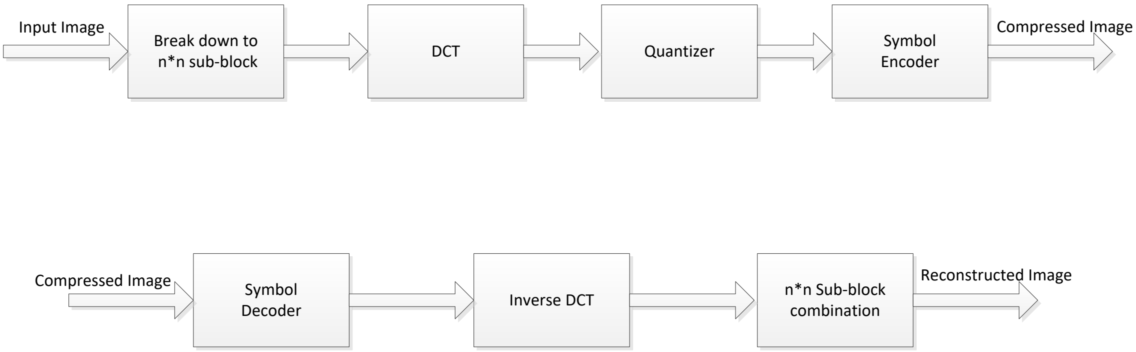

Figure 3.1: the Block Diagram of DCT based JPEG Transform

In the coding process, we first break down the image to n*n sub-blocks, normally 8*8 sub-blocks are chosen. The two-dimension DCT transform is used to get 64 coefficients for every sub-block. From the two-dimension DCT transform, we can get the 64 coefficients as follows:

From the two-dimension DCT transform, we can get the 64 coefficients as follows:

$$F(u,v) = \sum_{x=0}^{7} \sum_{y=0}^{7} f(x,y) a_u \cos\frac{(2x+1)u\pi}{16} \ a_v \ \cos\frac{(2y+1)v\pi}{16} \tag{3.2}$$

$$u = 0,1, \ldots 7; \qquad v = 0,1, \ldots 7$$

$$a_u = \begin{cases} \frac{1}{\sqrt{8}} \ , & u = 0 \\ \sqrt{\frac{2}{8}} \ , & u > 0 \end{cases} ; \qquad a_v = \begin{cases} \frac{1}{\sqrt{8}} \ , & v = 0 \\ \sqrt{\frac{2}{8}} \ , & v > 0 \end{cases}$$

Although the application of the DCT formulas would require O(N2) operations, it is possible to compute the same thing with only O(N log N) complexity. This method is known as fast cosine transform (FCT) algorithms. In this way, by paying the cost of more additions, we get speed faster since the CPUs are more excellent to do additions. It makes the DCT become applicable in real image compression processes. Due to good compression ratio, DCT get efficient memory utilization and it is widely used in practice.

In fact, the DCT transform does not compress the images, but the quantization process brings errors into the system. The quantization process is to divide the 64 coefficients by the quantization matrix and round the results into integers. We know this process will incur errors into the systems.

Let us assume the quantization matrix is Q[u,v], If the quantization errors are expressed as e[u,v], then the quantitated coefficient matrix is:

$$F_Q[u,v] = \frac{F[u,v]}{Q[u,v]} + e[u,v] \qquad (3.3)$$

In order to compress the image, some high frequency information is discarded since human eye are not sensitive to high frequency information, but this process also incur errors into the systems.

It is obvious that the errors will distribute inside the whole rebuilt image. Every sub-block bring different errors since they are calculated independently, hence the correlation between different sub-blocks are destroyed. This leads to the block artifacts effect. For the images, higher compression ratio leads to more severe block artifacts. Since the block artifacts effect comes from the DCT transform, it becomes the major disadvantage of the JPEG.

In order to show the results of DCT transform, we first simulate the DCT compression effects. We use an ‘autumn’ image to be our experiment object. Inside our code, we use different bit rates to compress the image.

Figure 3.2 and Figure 3.3 are both run with the scheme of 8*8 sub-blocks. The compressed images are more and more blur, and the block artifacts effect shows up more and more severely.

Figure 3.2 shows the original image and some DCT transformed images with light compression. Since the qualities of the compressed images are falling down little by little, the details are shown clearly and the qualities are satisfying for our eyes. The DCT transform appears to be very strong with light compression.

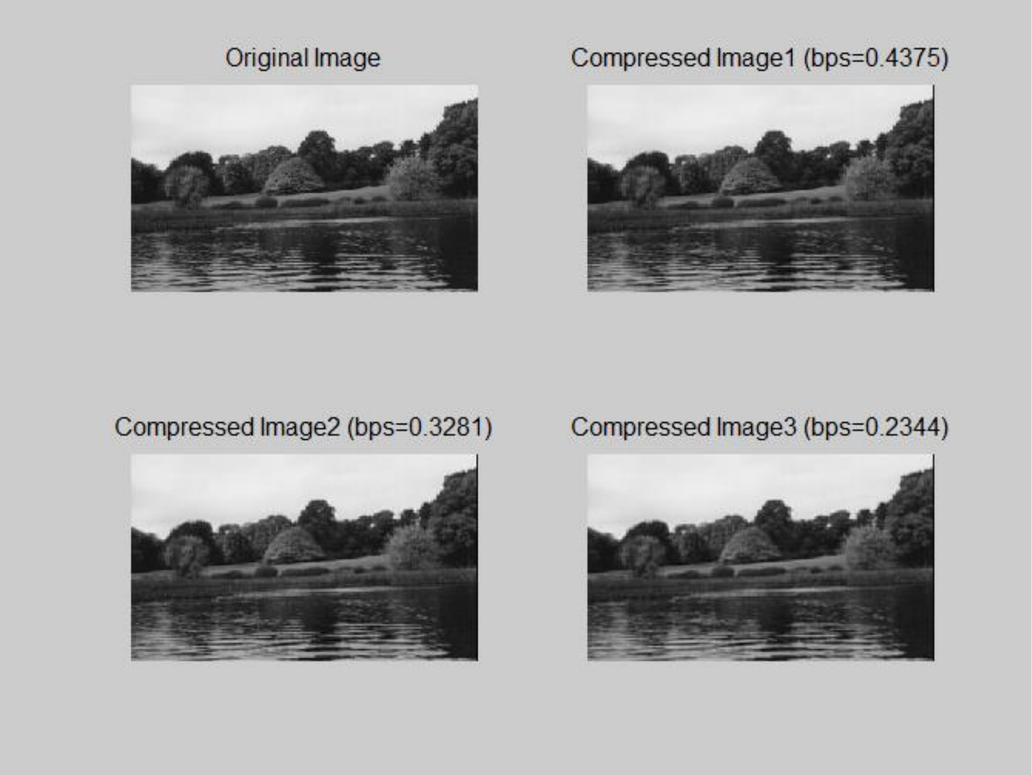
Original Image
Compressed Image1 (bps=0.4375)
Compressed Image2 (bps=0.3281)
Compressed Image3 (bps=0.2344)

Figure 3.2. DCT Transform –Autumn 1

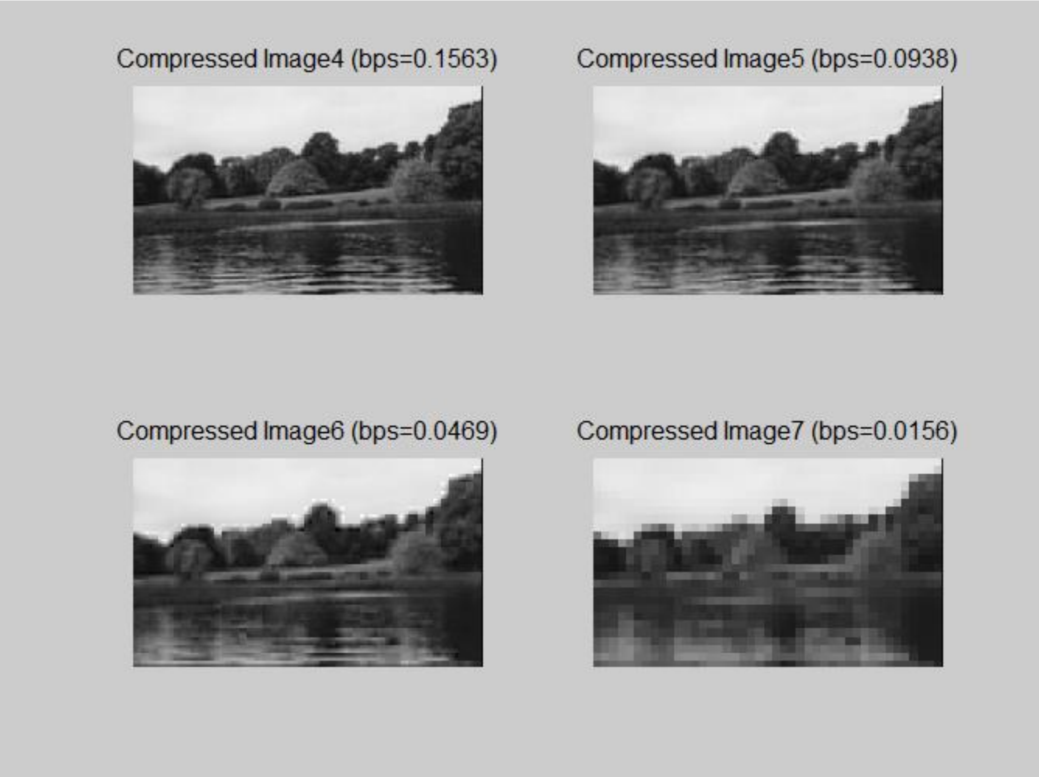
Compressed Image4 (bps=0.1563)
Compressed Image5 (bps=0.0938)
Compressed Image6 (bps=0.0469)
Compressed Image7 (bps=0.0156)

Figure 3.3. DCT Transform – Autumn 2

Figure 3.3 shows the DCT transformed images with lower bit rate than that of Figure 3.2. Inside this Figure, the block artifacts become more and more obvious, and it is not easy to view the details inside the image.

3.2 Background

In order to reduce the block artifacts with minimal change of the original coding system, many researchers have tried different methods. One common method to deal with images is to use filters to filter the received images. Among the filters, median filtered is a kind of non-linear filter and is considered suitable to eliminate the random noise [42]. Median filters are widely used in digital image processing. The idea of the median filter is to run through the signal pixel by pixel, replacing each pixel with the median value of neighbouring pixels. The pattern of neighbours is called the "window". For image signals, the windows are normally chosen as 3*3 pixels boxes, other shapes may be chosen. Another type of widely used filter is adaptive filter [43]. A common used adaptive filter is wiener filter. Wiener filter is a filter used to produce an estimate of a desired or target random process by linear filtering of an observed noisy process. It minimizes the mean square error between the estimated random process and the desired process.

Other methods are used to reduce the block artifacts effect. In [44], the authors proposed a hybrid method which acts both in the frequency and the spatial domains, to enhance the visual result of the reconstructed image and reduce the blocking artifacts. In [45], a semi-local approximation scheme to large-scale Gaussian processed was proposed. This allows learning of task-specific image enhancements from example images without reducing quality. Similar knowledge-based algorithms also include simultaneously removed JPEG blocking artifacts and recovers skin features [46]. On the other hand, some researchers use the property that the original pixel levels in the same block provide continuity and the correlation between the neighbouring blocks to reduce the discontinuity of the pixels across the boundaries [47]. [48] Proposes to remove artificial edges with a one-time nonlinear smoothing. But the shortcoming for all these methods is that they are very complex, which will consume a lot of computer CPU and memory resources. Note that the Urban Surveillance Systems accumulate huge number of images every second, it is impossible to deal with every image with long time process. It is important to propose very simple but effective methods to solve the block artifacts effect. The best method should be kept at a single step after receiving the already affected image. In this way, all the invested hardware and software may be kept and the cost will be kept low. This motives us to propose the following new simple but effective method.

3.3 Our Proposed Method and Results

We notice that the block artifacts bring some sharp changes between the vertical and horizontal adjacent blocks. The situation is shown in Figure 3.4. Hence our idea is to only deal with the sudden changes. In order to keep as much as the original image, our proposed method is to smooth only the connection edges of the vertical and horizontal adjacent of the blocks while keeping other parts of the image unchanged.

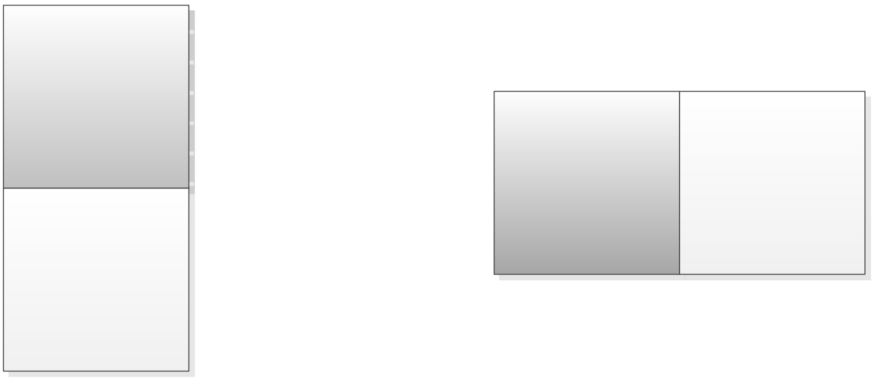

Figure 3.4. Vertical and horizontal adjacent blocks

Using an example of horizontal direction, our proposed method is illustrated as follows:

First we calculate the difference between the two adjacent blocks, which can be written as $\Delta=abs(x(i)-x(i-1))$, where x(i) and x(i-1) are the grey values at the right and left of the line of the sudden change. In order to make the change smooth, we divide the Δ by 3. The last step is to adjust the grey values of the right and left of the sudden change line by adding or subtracting the value by $\Delta/3$. In this way, the sudden change is smoothed while keeping most pixels unchanged, hence to alleviate the block artifacts. The flow chart of our proposed method is shown in Figure 3.5 (shown for horizontal direction only).

We simulate the effect of our proposed method and compare with the two kinds of widely used filters: the median filter and wiener filter. As we already know, at those lightly compressed images, the block artifacts are not obvious; hence we only need to deal with the poor quality received images, which are the images with deep compression. The results are shown in Figures 3.6, 3.7 and 3.8. Inside these figures, the compression ratio means the size ratio of the compressed image file and the original image file.

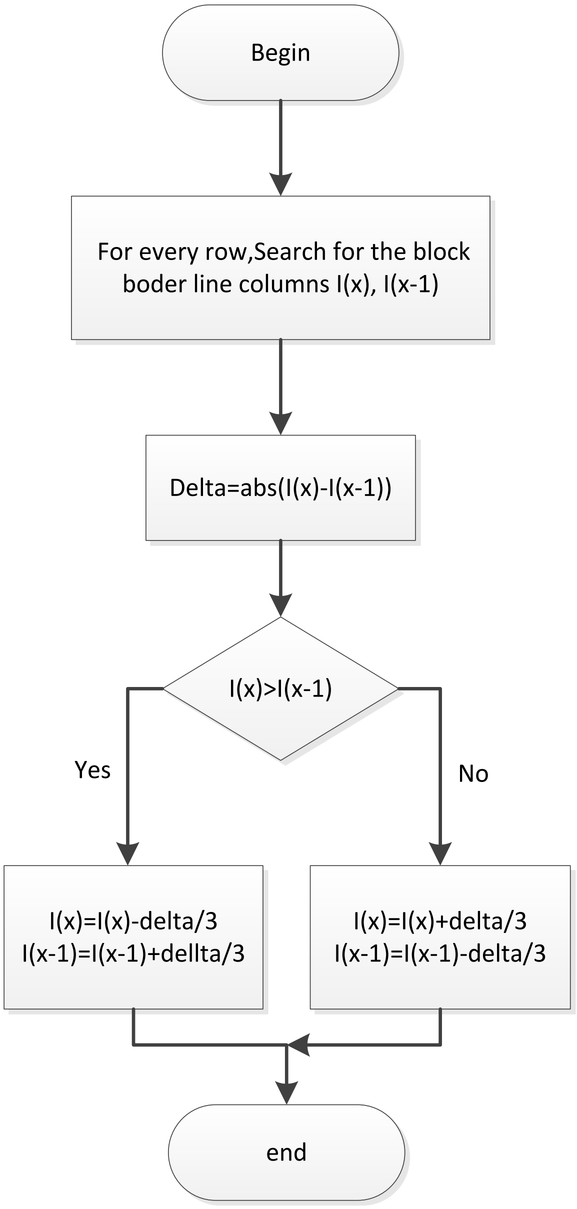

Figure 3.5. Flow chart of our proposed method (for horizontal line)

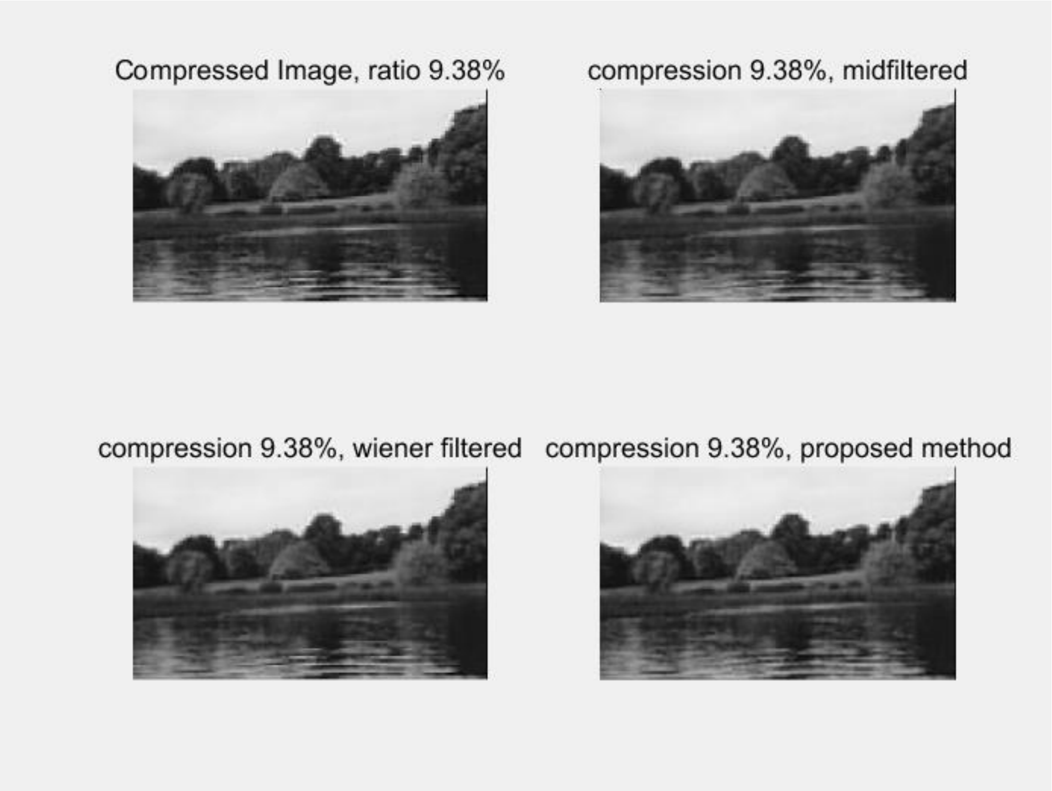
Compressed Image, ratio 9.38%
compression 9.38%, midfiltered
compression 9.38%, wiener filtered
compression 9.38%, proposed method

Figure 3.6. The effects of different methods (compression ratio is 9.38%)

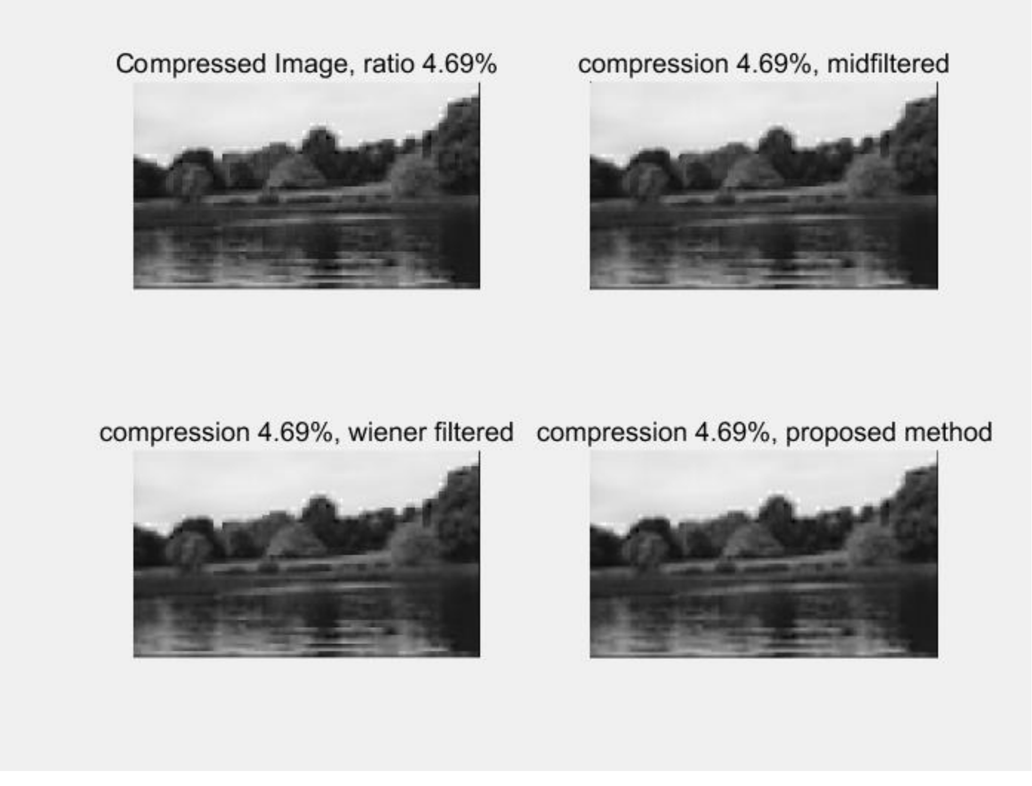
Compressed Image, ratio 4.69%
compression 4.69%, midfiltered
compression 4.69%, wiener filtered
compression 4.69%, proposed method

Figure 3.7. The effects of different methods (compression ratio is 4.69%)

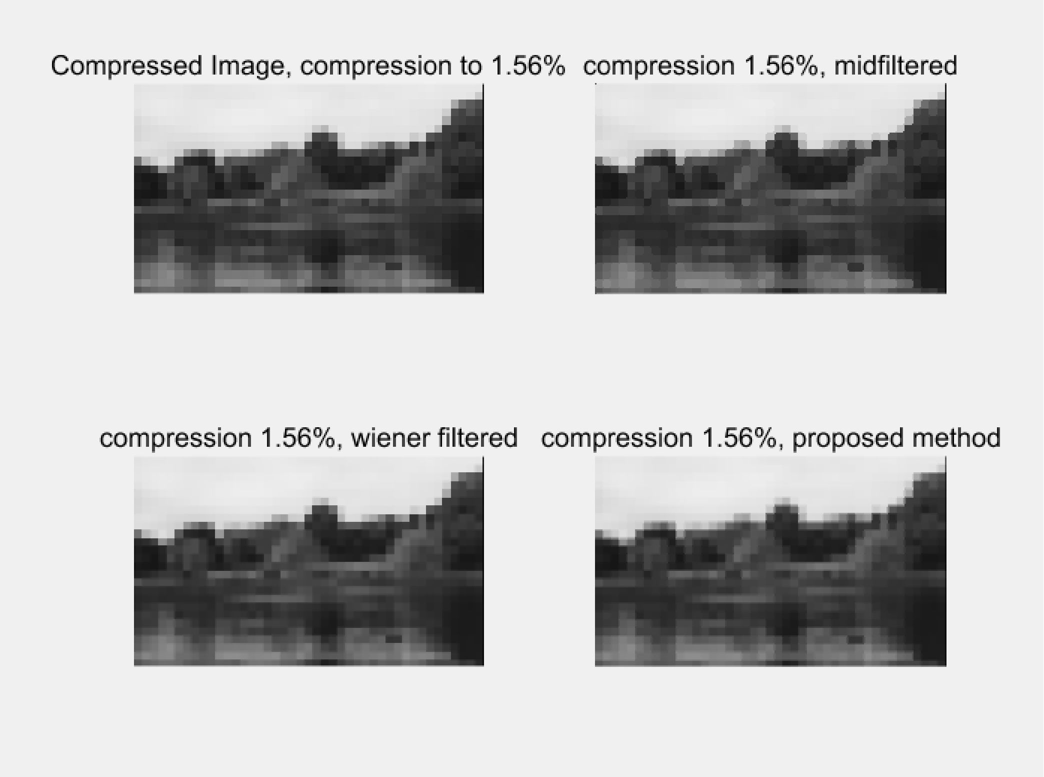
Compressed Image, compression to 1.56%
compression 1.56%, midfiltered
compression 1.56%, wiener filtered
compression 1.56%, proposed method

Figure 3.8. The effects of different methods (compression ratio is 1.56%)

Since it is not easy to compare the effects by our eyes, we need some actual numerical results. Hence, we calculated the Peak Signal to Noise Ratio (PSNR) of all the different methods. PSNR represents a measure of the peak error; it is a widely used metric to compare image compression quality. With higher PSNR value, the quality of an image is considered better. Here we show the PSNR results in Table 3.1.

Bit rate	Unfiltered	Median filtered	Wiener filtered	Our method
0.0938	24.9777	25.0860	25.0572	**25.2193**
0.0469	23.1011	23.2140	23.2074	**23.3213**
0.0156	20.9932	21.0514	21.1097	**21.4136**

Table 3.1. The PSNR of different situations (Autumn)

This table shows that, when the images are heavily compressed, the filters and our proposed method do effect to smooth the sudden changes caused by block artifacts, hence improve the quality of the images. But our proposed method got better results than these widely used filters. Our method results in higher PSNRs, which means better qualities.

In order to test our proposed method, we simulate the effects of our proposed method and the two kinds of widely used filters on another famous image: cameraman. The results are shown in Figures 3.9, 3.10 and 3.11. We also calculated the PSNRs of all the different methods. Here we show the PSNR results in Table 3.2.

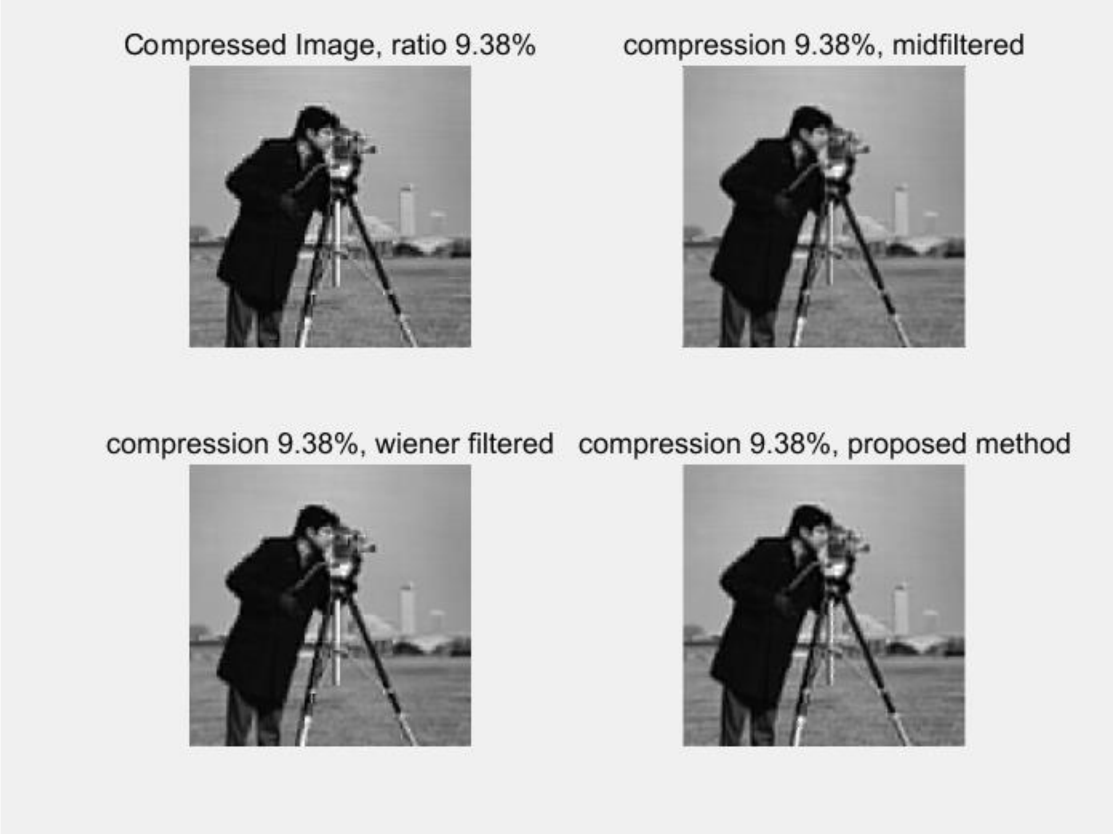
Compressed Image, ratio 9.38%
compression 9.38%, midfiltered
compression 9.38%, wiener filtered
compression 9.38%, proposed method

Figure 3.9. The effects of different methods (compression ratio is 9.38%)

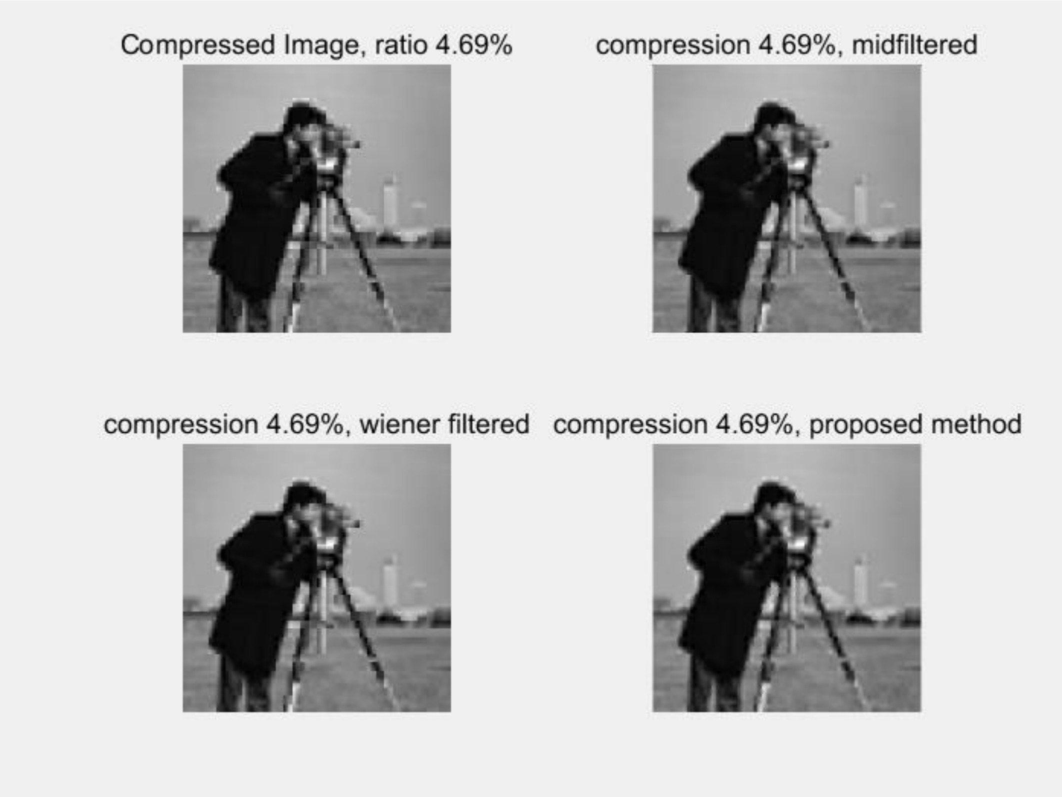
Compressed Image, ratio 4.69%
compression 4.69%, midfiltered
compression 4.69%, wiener filtered
compression 4.69%, proposed method

Figure 3.10. The effects of different methods (compression ratio is 4.69%)

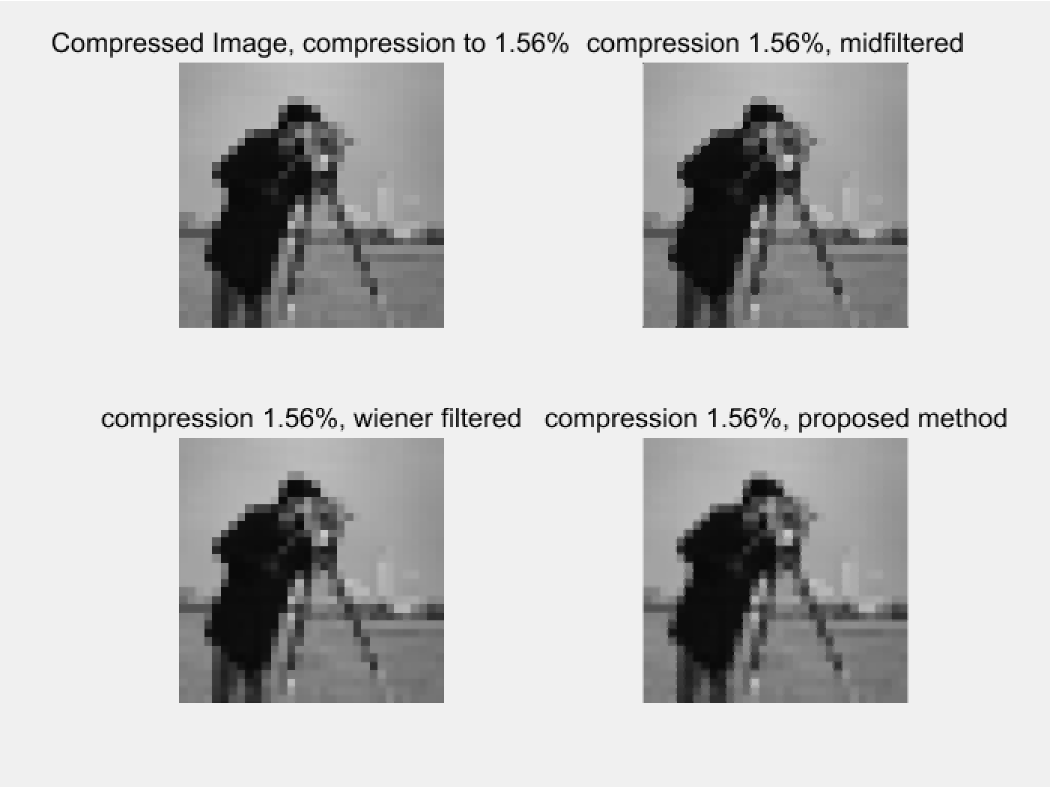
Compressed Image, compression to 1.56%
compression 1.56%, midfiltered
compression 1.56%, wiener filtered
compression 1.56%, proposed method

Figure 3.11. The effects of different methods (compression ratio is 1.56%)

Bit rate	Unfiltered	Median filtered	Wiener filtered	Our method
0.0938	23.7897	23.9410	23.8822	**23.9571**
0.0469	21.9168	22.1191	22.0286	**22.1938**
0.0156	19.4785	19.5574	19.5837	**19.8362**

Table 3.2. The PSNR of different situations (Cameraman)

Table 3.2 again shows that our proposed method results in better quality. The main benefit of our proposed method is that the comparably better PSNR results are gotten with very simple calculations. Considering that normal filters need to calculate every pixel for an M*N image while our proposed method only need to calculate the pixels at the block borders, our method only need a small fraction of the CPU calculation source comparing with a normal image filter.

3.4 Summary

The Urban Surveillance Systems are facing the problem of huge amount of video data, and the compression technique is the most fundamental and important to deal with the data. Although the DCT based JPEG standard are widely used, it encounters problems such as block artifacts. In this chapter, in order to reduce the block artifacts, we proposed a very simple but effective method. The simulation results show that our proposed method results in better quality than widely used filters while consuming much less computer CPU resources. In this way, we could reduce the visual block artifacts of DCT compressed images fast for Urban Surveillance Systems.

CHAPTER 4

IOT-DRIVEN AUTOMATED OBJECT DETECTION ALGORITHM FOR URBAN SURVEILLANCE SYSTEMS IN SMART CITIES

Part of this chapter was published as a research paper below:

L. Hu, Q. Ni, “IoT-Driven Automated Object Detection Algorithm for Urban Surveillance Systems in Smart Cities”, IEEE Internet of Things Journal, Volume: 5, Issue: 2, Pages: 747-754, April 2018.

Automated object detection algorithm is an important research challenge in intelligent urban surveillance systems for IoT and smart cities applications. In particular, smart vehicle license plate recognition (VLPR) and vehicle detection are recognized as core research issues of these IoT-driven intelligent urban surveillance systems. They are key techniques in most of the traffic related IoT applications, such as road traffic real-time monitoring, security control of restricted areas, automatic parking access control, searching stolen vehicles, etc. In this chapter, we propose a novel unified method of automated object detection for urban surveillance systems. We use this novel method to determine and pick out the highest energy frequency areas of the images from the digital camera imaging sensors, that is, either to pick the vehicle license plates or the vehicles out from the images. Our proposed method can not only help to detect object vehicles rapidly and accurately, but also can be used to reduce big data volume needed to be stored in urban surveillance systems.

4.1 Introduction

Smart transportation and urban surveillance systems are important internet of things (IoT) applications for smart cities [49] [50]. In these smart transportation and urban surveillance applications, cameras/imaging sensors are commonly installed to automatically detect and identify potential vehicles/cars through automated object detection methods. Usually, such automated object detection methods demand high-complexity image/data processing technologies and algorithms. Hence, the design of low-complexity automated object detection algorithms becomes an important topic in urban surveillance systems. Among these researches, both vehicle license plate recognition (VLPR) and vehicle recognition are hot research topics worldwide, which can be applied to many IoT applications, such as road traffic data collection/monitoring, automatic parking charging and access control, and searching stolen vehicles.

It is known that a license plate number is a unique identification of a vehicle. Specifically, the license plate recognition, i.e. the extraction of a license plate region from an image, is the key module in a VLPR system [51], which influences the accuracy of the VLPR systems significantly. Different algorithms have been proposed for identifying a vehicle license plate using image processing [52]. One typical way is vertical edge matching [53]. The idea is to first locate the two vertical edges of a license plate, and hence to detect its four corners. In this way, the license plate can be extracted accurately. Using the contrast between the grayscale values, [54] proposed a vertical edge based license plate recognition method.

Another technology is morphology based license plate detection. This method is to extract important features of contrast as guidance to search the license plates [55]. In [56], to extract potential text information from the image, a method is proposed using adaptive threshold, fractal filter and morphological analysis. In [57] and [58], edge statistics in combination with morphological approaches are proposed to eliminate the undesired edges in the images.

Color based methods are also attempted which make use of the colors of the vehicle license plate. In [59], a color based method combined with the texture characteristics is proposed to try to detect license plate from the color image. In [60] and [61], the combination of edge information and plate color are utilized to identify the vehicle license plates.

Based on neural network techniques, other recognition methods of vehicle license plates are proposed. These methods are designed to train classifiers to offer a proper response to the license plate images. In [62], the authors apply genetic algorithm (GA) to the training process and combine the statistic features together with structure features. In [63], a vehicle license plate detection method using neural network approaches is proposed. The proposed scheme utilizes a neural network chip named as CogniMem to detect the vehicle license plates. In [64], the authors propose a method using wavelet transform technique to decompose the images into different layers, and then utilize the low frequency images to combine with neural network technique.

While most approaches have attempted to deal with the VLPR with optical characteristics recognition, another method is proposed based on the horizontal and vertical frequency energy differences. In [65], the authors use Daubechies wavelet transform method to calculate the vertical and horizontal frequency energy curves, considering the frequency is the highest near the number plate.

Other researchers focus on the vehicle recognition. In [66], the authors propose a combination of the transfer subspace learning technique with the manifold learning approach to allow a more systematic search of tuning parameters for the purposes of cross-data and cross-domain electro-optical vehicle recognition. The authors in [67] propose a vehicle recognition and retrieval system using the so-called bag of words approach. In this way, the system may determine automatically the manufacturer and the type of the captured vehicle images. The authors in [68] present a vehicle recognition system prototype using magnetic sensor techniques where more complex hardware and processing are utilized.

However, all the above mentioned existing methods in the literature are still too complex which demand high computational processing. Furthermore, each method can only be used to recognize a vehicle license plate or a vehicle individually. None of them can be used as a unified method to detect both vehicle license places and vehicles.

In this chapter, we propose a novel, simple and unified method to search the objects by filtering out the vehicle and/or license plate images rapidly from the digital camera imaging sensors. We design a simple filter to effectively detect either vehicle license plates or vehicles, motivated by the observation that in the most cases the object is the highest energy frequency part of an image according to the character of vehicle or vehicle plate inside the images taken from roads. This process can be easily implemented in any urban surveillance systems in smart cities to pick out comparably important areas from the images captured by any camera/imaging sensors in urban environments, which will be a useful method not only to rapidly detect the important information but also to reduce the large data volume required to be stored because only those selected important (smaller) data will be stored as compared to huge raw data captured/generated from any cameras/imaging sensors 24 hours per day, 7 days a week and 365 days per year.

4.2 Our Proposed Algorithm

4.2.1 Pre-Treatment of the Image

As an instance, let us look at Figure 4.1. From this image, we need to pick out the car plate, which is the highest energy frequency area. Before formal treatment of the image, we carry a pre-treatment of the image. Firstly we convert the original RGB (R: red, G: green, B: blue) image into grey scale image since the color information is not needed in our method, and the size of the image file can be reduced to a lower level.

Since the human eye has different degrees of perception of different light, we use a general formula for image processing to convert the image:

$$A_{gl} = \frac{3A_r + 6A_g + A_b}{10}\,, \qquad (4.1)$$

where Ar denotes the red spectrum of the color image, Ag is the green spectrum of the color image, and Ab is the blue spectrum of the color image. The result Agl is the converted grey scale level. The grey scale image is shown in Figure 4.2.

In the second step, we calculate and obtain the gradient of the grey image. We consider the two-dimensional grey scale image as a two-dimensional matrix P, and calculate the numerical gradient of this matrix. Px corresponds to the differences in x (horizontal) direction. Py corresponds to the differences in y (vertical) direction. The definition to calculate the Px and Py are:

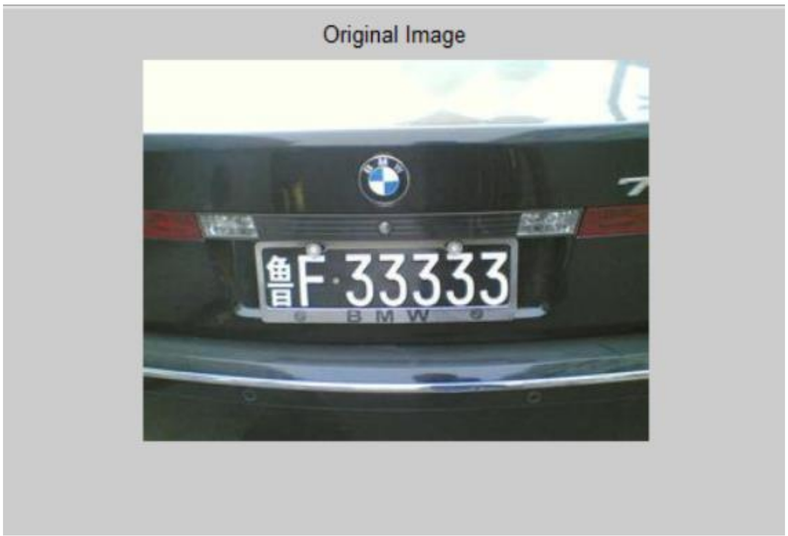

Figure 4.1. The original car image

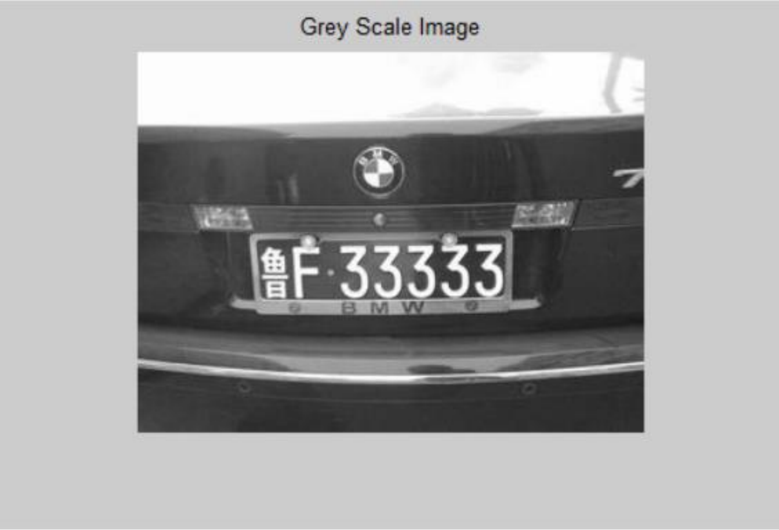

Figure 4.2. The grey scale image

$$\begin{cases} P_{x(i,j)} = I(i+1,j) - I(i,j) \\ P_{y(i,j)} = I(i,j+1) - I(i,j) \end{cases}, \tag{4.2}$$

where I is the grey scale value of the grey scale image, and (i,j) is the position of the pixel.

For every pixel, after getting the gradient of both the horizontal and vertical directions, we calculate the overall gradient of this pixel:

$$P = \sqrt{{P_x}^2 + {P_y}^2}\,. \tag{4.3}$$

The gradient result is shown in Figure 4.3. In Figure 4.3, some large areas of constant grey scale fade into darkness because of the slow movement of gradient. Now the car image has finished pre-treatment, the image has been well prepared to implement our method. The next step is how to pick out the car license plate area automatically.

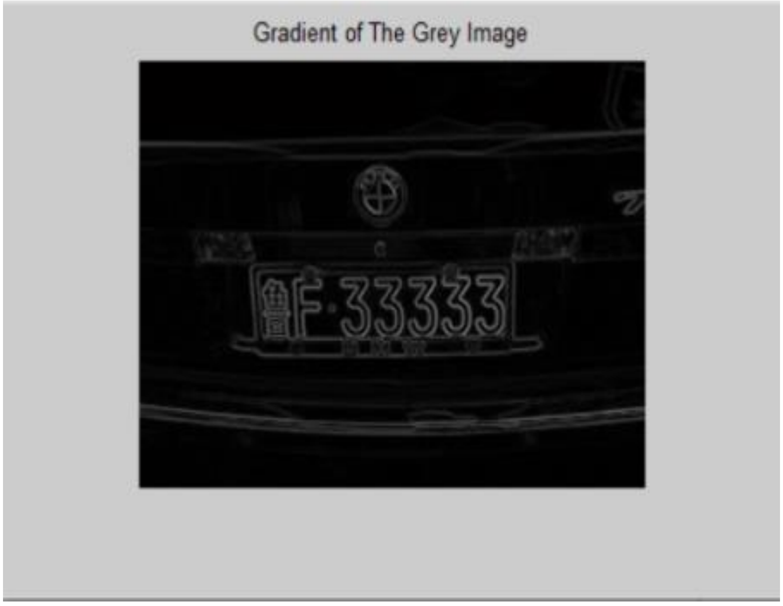

Figure 4.3. The gradient of the grey image

4.2.2 Design of Novel Filter

Our idea is to design a two-dimensional smart filter to pick out the object area of an image. We first investigate the characteristics of the object. We found that, within the object boundaries, the energy frequency is high, and the energy frequency curves down sharply outside the object (e.g. the car license plate) boundaries. This interesting characteristic inspires us to design a new two-dimensional filter to figure out the horizontal and perpendicular frequency energy curves, since the meaning of filtering is to calculate the intercorrelation of the filter and the image matrix. Mathematically, the definition of intercorrelation is a mutual relationship or connection between two or more things. The formula for the correlation coefficient is:

$$\mathrm{r} = \frac{\sum_1^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_1^n (x_i - \bar{x})^2}\sqrt{\sum_1^n (y_i - \bar{y})^2}}. \qquad (4.4)$$

Our objective is to implement a new filter to calculate the correlation coefficient for every pixel of the car image and obtain a new image with the car license plate area standing out. To achieve this, we investigate the function for intercorrelation which is expressed as follows:

$$f(x)^{\circ}g(x) = f(x)^{*} * g(x) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f^{*}(u,v)g(x-u,y-v)dudv. \quad (4.5)$$

The result of the intercorrelation operation reflects the measure of similarity between the two functions *f(x)* and *g(x).* This motivates us to utilize the pulse function δ(x) which holds an important characteristic as follows:

$$\int_{-\infty}^{\infty} \delta(x-a)\varphi(x)dx = \varphi(a). \quad (4.6)$$

That is, the function δ(x-a) can be used to pick out another function φ(x) at the x-axis when x=a. We use this property to pick out the highest energy frequency area of a car image.

Specifically, we define line arrays to simulate the pulse function δ(x). After the filtering process, the result will be shown in a new image.

As for our proposal, a perpendicular line array filter (in the array, all the numbers are set as 1's) is used to filter the pre-treated image in horizontal direction, and another horizontal line array filter (in the array, all the numbers are set as 1's) is used to filter the pre-treated image in perpendicular direction. The diagrammatic sketch is shown in Figure 4.4.

We first filter the pre-treated image in the horizontal direction, since the vehicle plate (or a vehicle) normally holds larger size in this direction. Hence, it is easier to eliminate noise in this direction. The results are shown in Figure 4.5. In this way, the high energy frequency areas are shown off by brighter lines. In order to get the intuitive impression, we plot the one-dimensional brightness values for straightforward view by choosing the middle horizontal array line of the filtered image. The values obtained in the x-axis are shown in Figure 4.6.

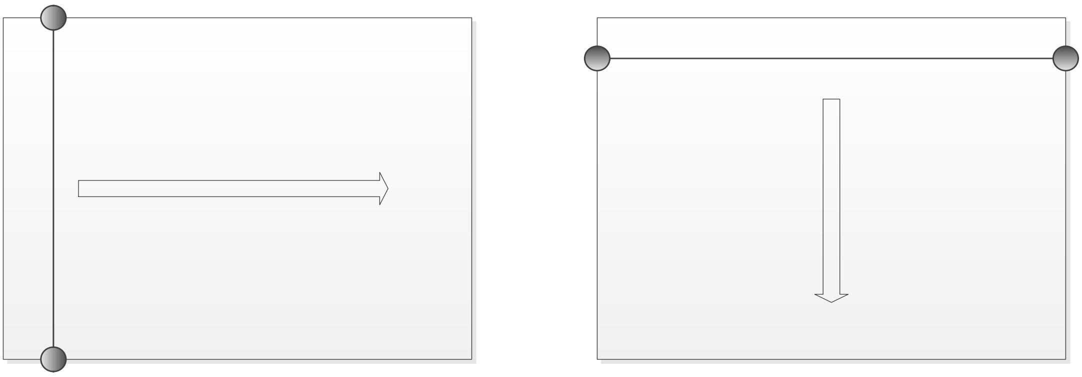

Figure 4.4. Line array filters to filter the pre-treated image

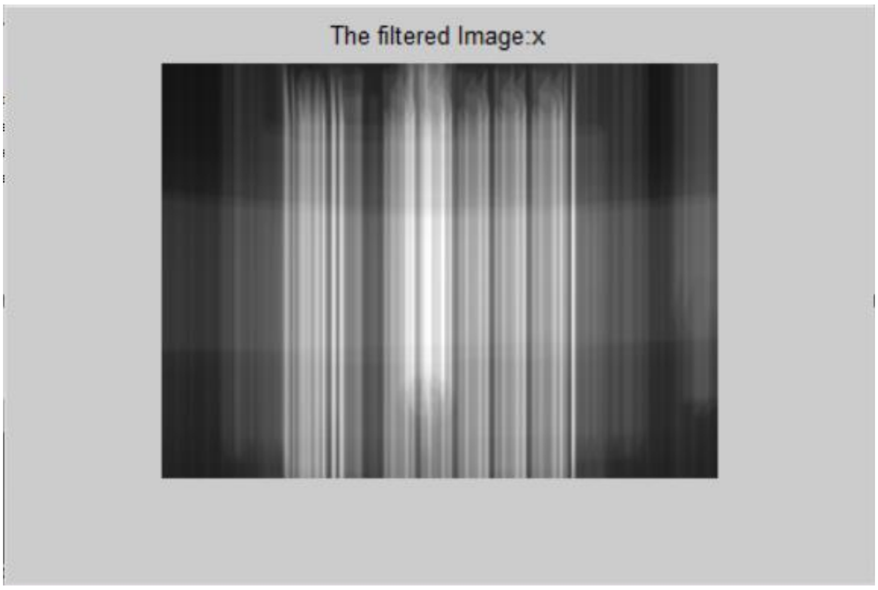

Figure 4.5. The horizontal filtered image

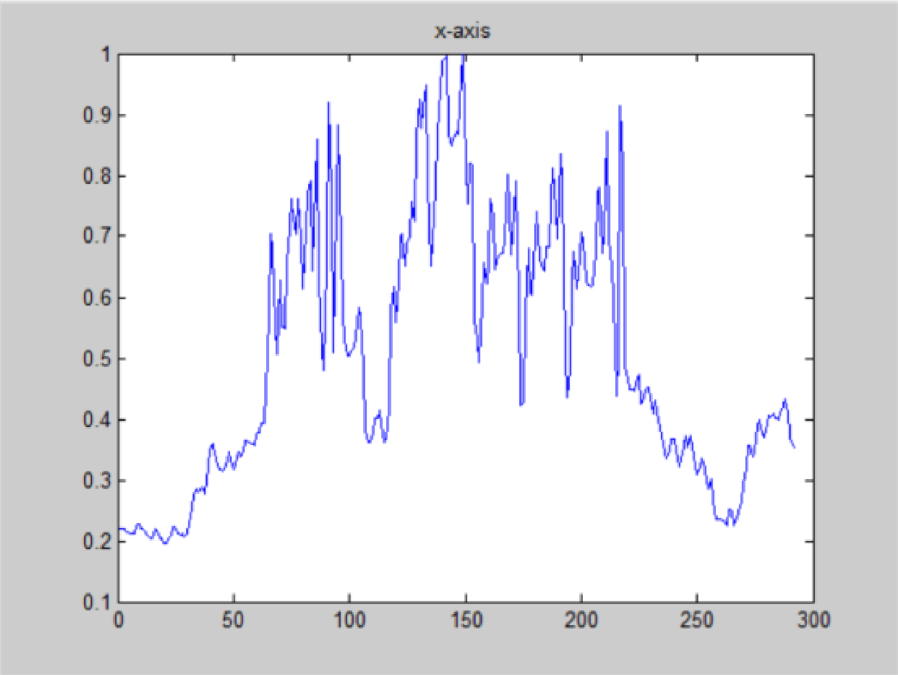

Figure 4.6. The horizontal axis's brightness values

From Figures 4.5 and 4.6, we find that the brightness values show off the plate area in the horizontal direction roughly, with the higher value representing high energy frequency. The shortcoming is the noise, which makes how to detect the threshold difficult. In order to solve the problem of the noise, we need to improve our filter to smooth the noise while keeping this filter simple and effective.

4.2.3 Improvement of the Filter

By researching the filtered image, we find that the noise comes from the thin lines of the filter. In order to merge the sharp edges between the lines, one way is to expand the width of the filter. On the other side, the filter width can't be too wide, since it will lead to blurred filtered image. In the horizontal direction, we expand the original line array filter into 30 lines of 1's, that is, to expand the filtered line into a matrix. In this way, the filter becomes a long and narrow rectangle. This perpendicular matrix (all the numbers are set as 1's) is used to filter the image in horizontal way. The diagrammatic sketch of this filter is shown in Figure 4.7.

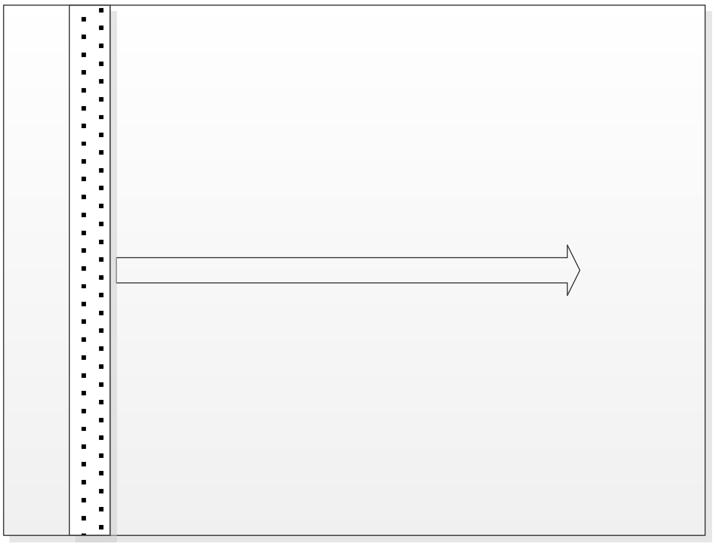

Figure 4.7. Matrix filter in horizontal direction

We use this improved filter to filter the pre-treated image in horizontal direction; the results are shown in Figure 4.8. In this direction, the high frequencies areas again are figured out by brighter areas.

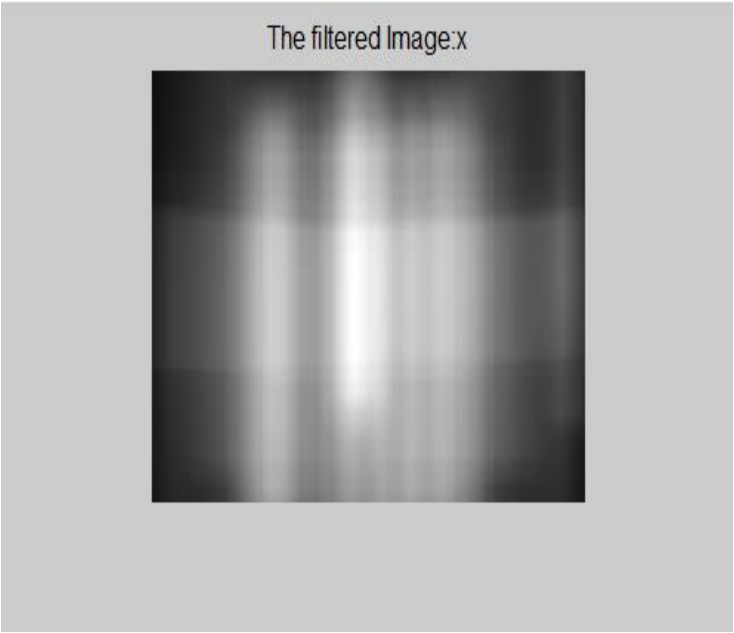

Figure 4.8. The horizontal filtered image

In order to get the intuitive impression, we plot the one-dimensional figure. By choosing the middle horizontal brightness values of the filtered image, we plot the results in Figure 4.9. The results are smoothed and noise is reduced while the main shape remains. We found that by changing the width of the matrix filter, we can change the smoothness of the horizontal axis's brightness values. The filter width parameter can be adjusted according to the object character.

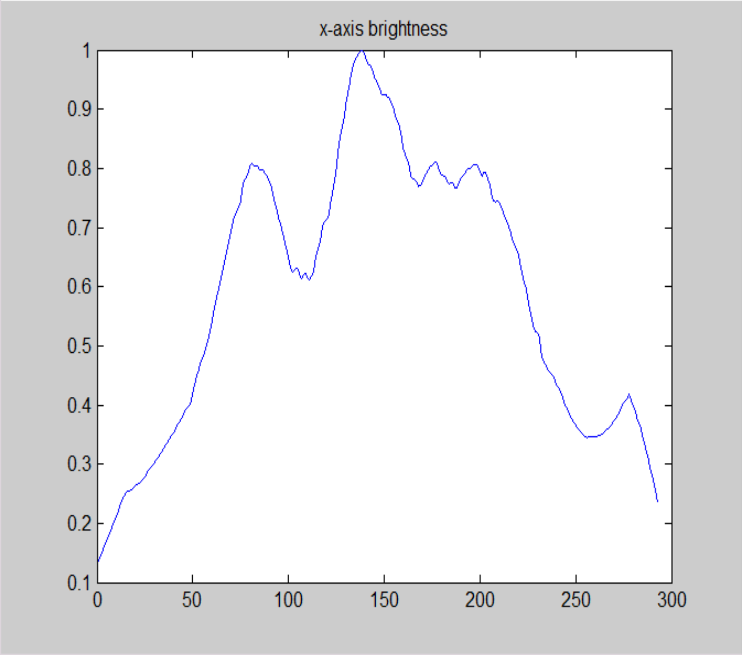

Figure 4.9. The horizontal axis's brightness values

4.2.4 Object Area in the Horizontal Direction

Since the car image has been filtered perfectly, to cut out the plate area is not difficult now. As for the horizontal axis, the whole plate area is the brightest area. According to this feature, we only need to scan the horizontal axis values from left to right and simply choose the threshold brightness of 0.5. In order to eliminate the interruption of unwanted small brightness, such as the edges of the car, we check the picked width between the thresholds. Since the width of the car license plate is within an already-known area, we can cut out the car license plate in x-axis correctly.

We know that the width of the filter matrix brings some blur and expands the width of the car plate, and the filter scans the image from left to right. Hence we only need to gently adjust the final results by shrinking the detected area and move the results a little left.

After obtaining the x-axis value, we cut this horizontal area from the original image, and the result is shown in Figure 4.10.

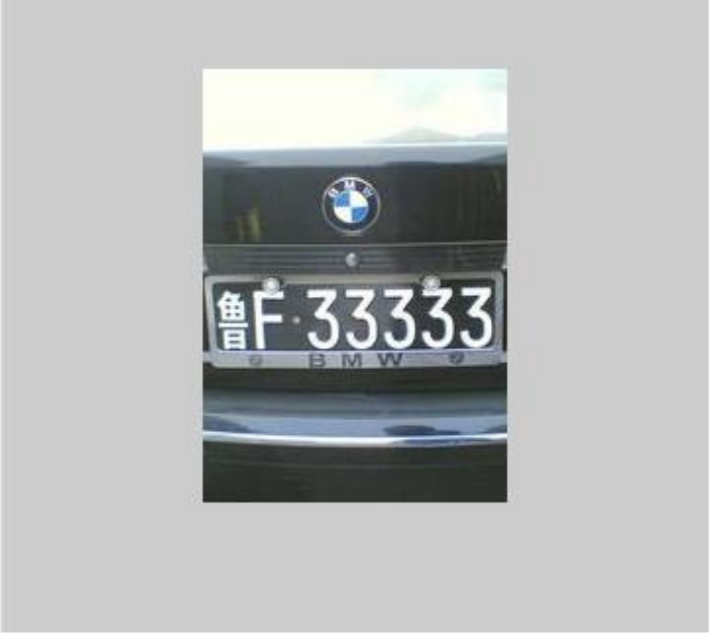

Figure 4.10. The object area in horizontal direction

4.2.5 Use Filter to Detect the Perpendicular Direction Area

Similar to the horizontal direction, we use matrix filter in the perpendicular position. Since the car license plate area in horizontal direction has already been picked out, in order to avoid the interference of unwanted noise, we only need to filter the pre-treated image in the area shown in Figure 4.11. We notice that in the perpendicular direction, there is no break between the characters like the horizontal direction; hence the height of the filter does not need to be big. Here we choose the height number as 8 to filter the object image. The diagrammatic sketch of this filter is shown in Figure 4.11.

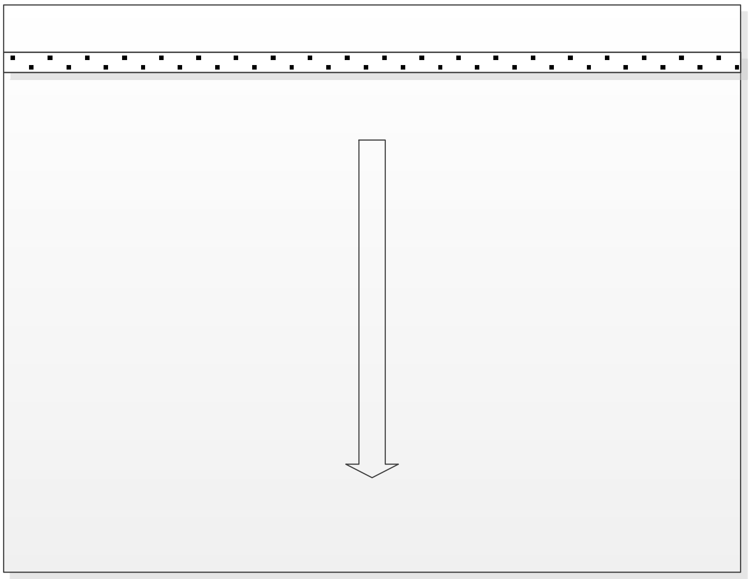

Figure 4.11. Matrix filter in perpendicular direction

After filtering the pre-treated image within the area shown in Figure 4.10, the perpendicular results are shown in Figure 4.12. In this direction, the high frequencies areas are shown out by brighter areas.

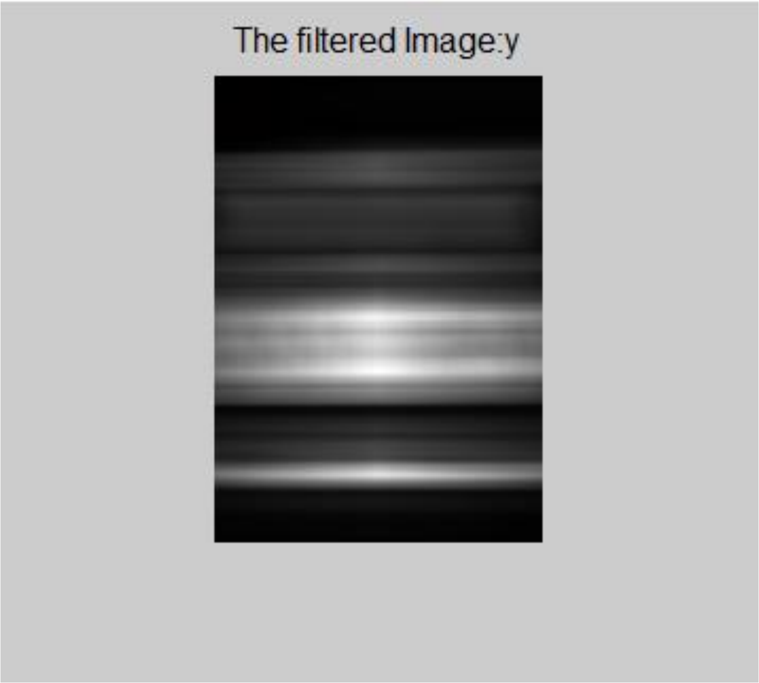

Figure 4.12. The perpendicular filtered image

In order to get the intuitive impression, we plot the one-dimensional figure for straightforward view as well. By choosing the middle perpendicular brightness values of the filtered image, we plot the results in Figure 4.13. Because the width of the filter is small, the results are smoothed and noise is reduced while the main shape remains.

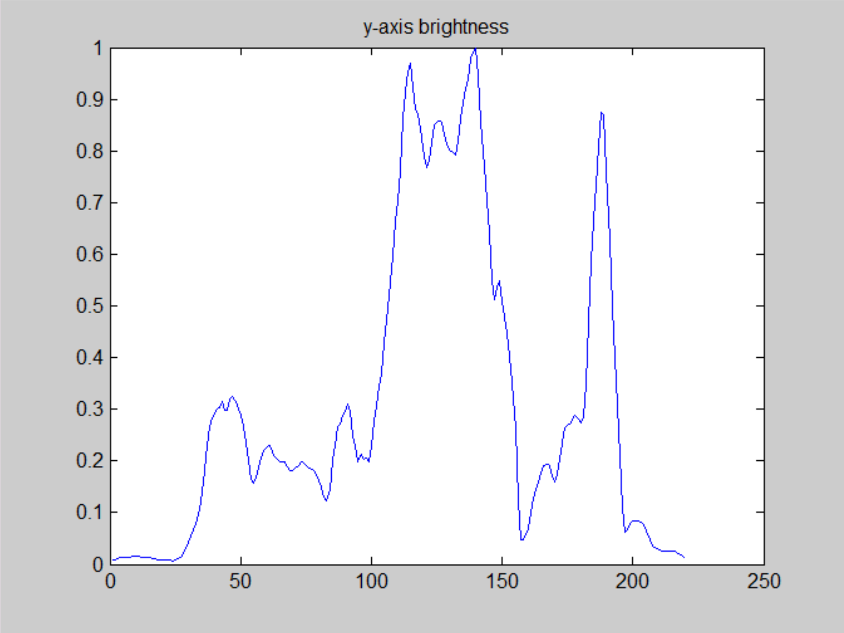

Figure 4.13. The perpendicular axis's brightness values

Now we need to cut out the plate area in the perpendicular direction. As for the perpendicular axis, the whole plate area is the brightest area although there are some interruptions of noise coming from the edges of the car, with the noise brightness normally shown as narrow pulse. According to this feature, we only need to scan the perpendicular axis's brightness values from left to right and simply choose the threshold value at 0.5. In the scan process, we check the picked width between the thresholds. Since the height of the car license plate is within an already-known area, we can cut out the car license plate in perpendicular direction correctly.

We then gently adjust the final results by shrinking the detected area and move the results a little up. After getting the four edge value, we cut this area from the original car image, and the result is shown in Figure 4.14.

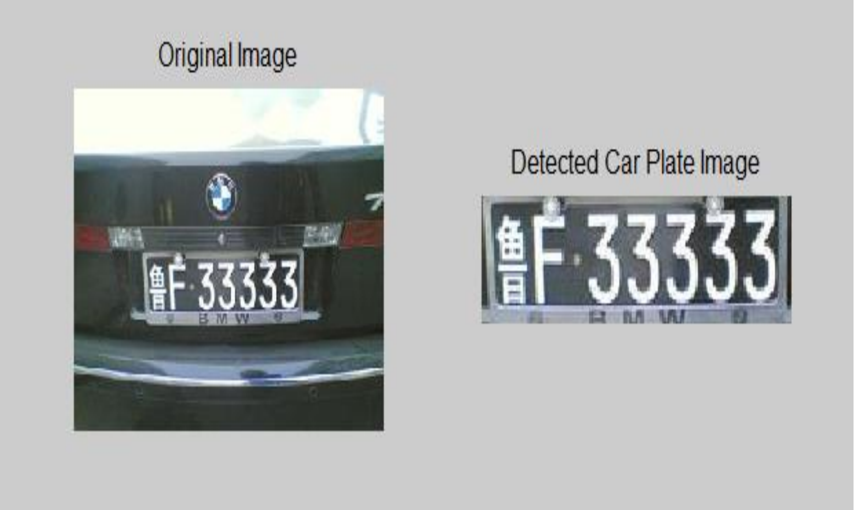

Figure 4.14. The detected car license plate-1

The result shows the correctly picked car license plate. Consider that we only use a simple filter to get the result; this demonstrates that our method is effective and practical.

4.3 The Flow Chart of Our Proposed Method

The flow chart of our proposed method is shown in Figure 4.15. It is a simple method to pick out the highest energy frequency area of a vehicle image.

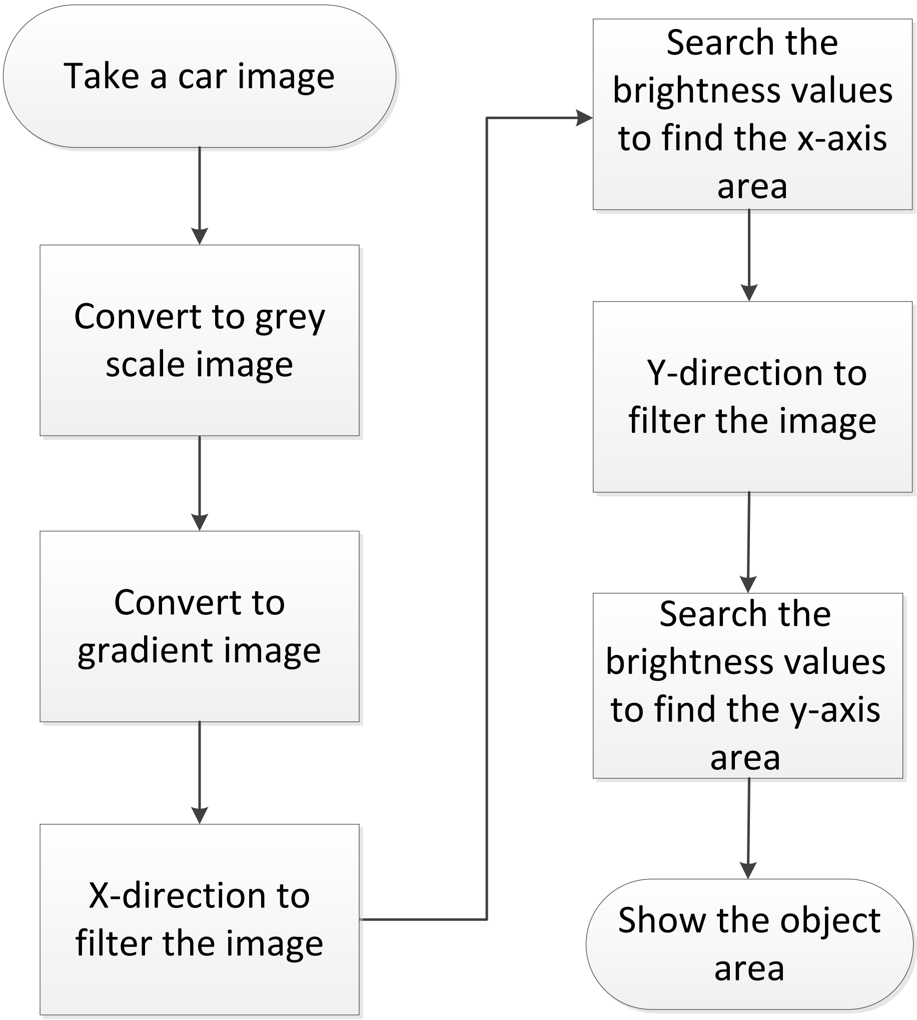
Take a car image
Convert to grey scale image
Convert to gradient image
X-direction to filter the image
Search the brightness values to find the x-axis area
Y-direction to filter the image
Search the brightness values to find the y-axis area
Show the object area

Figure 4.15. The process of our proposed method

4.4 Experimental Results of Our Proposed Filters

4.4.1 Use Our Filter to Detect Vehicle License Plates

In this section, we test our filter to see how it works on more car image with license plates. Our objects are the vehicle license plates. The car images are come from internet. Some results are shown in Figures 4.16 and 4.17 with the same parameter settings.

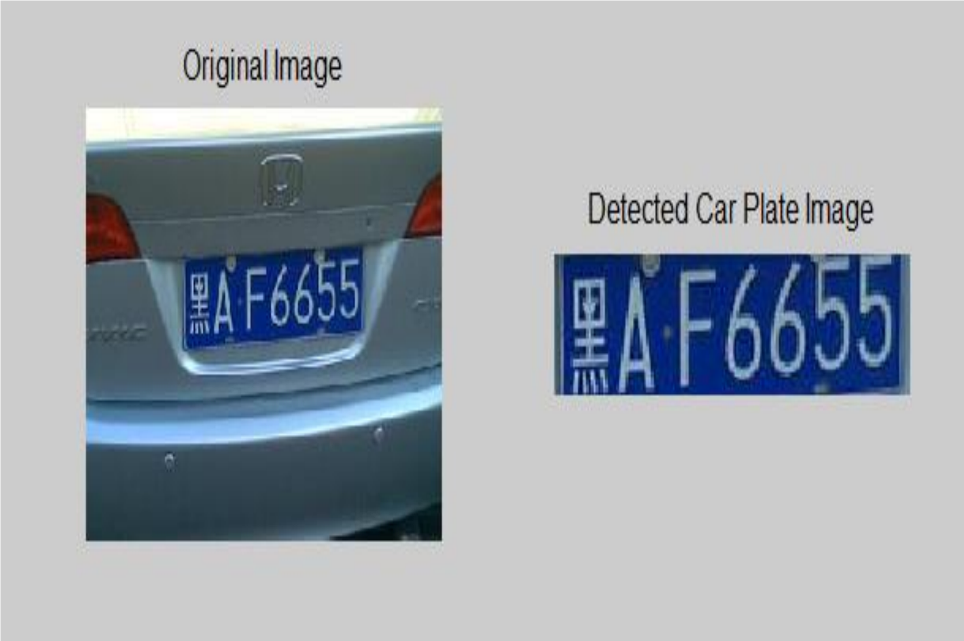

Figure 4.16. The detected car license plate-2

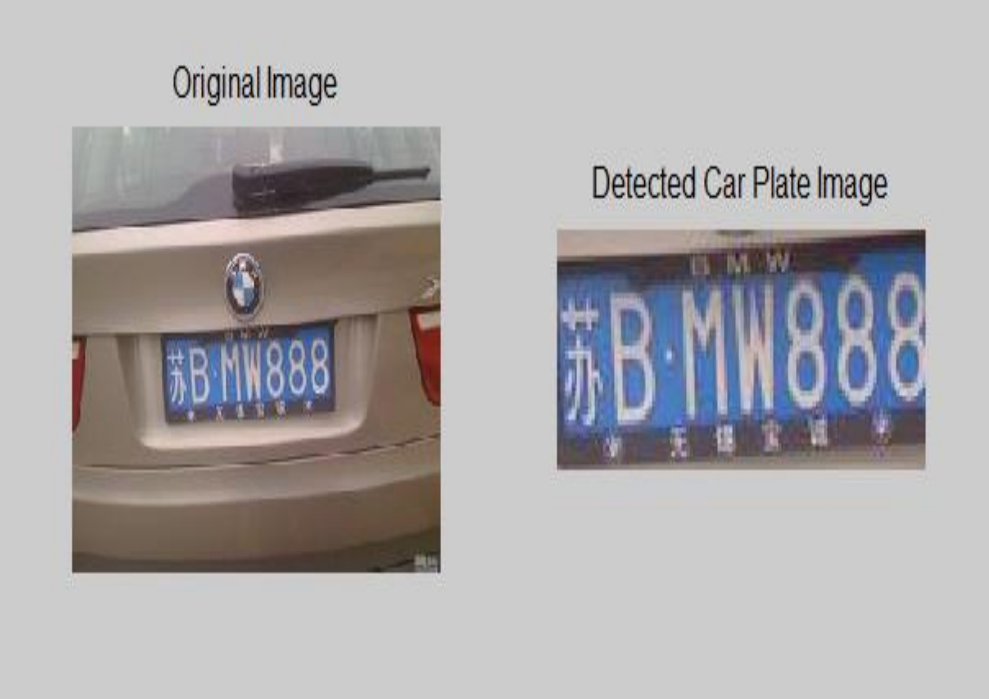

Figure 4.17. The detected car license plate-3

In many situations, the car license plates are not such big as shown in Figures 4.15, 4.16 and 4.17. They may appear like in Figure 4.18, a whole car with a smaller car license plate:

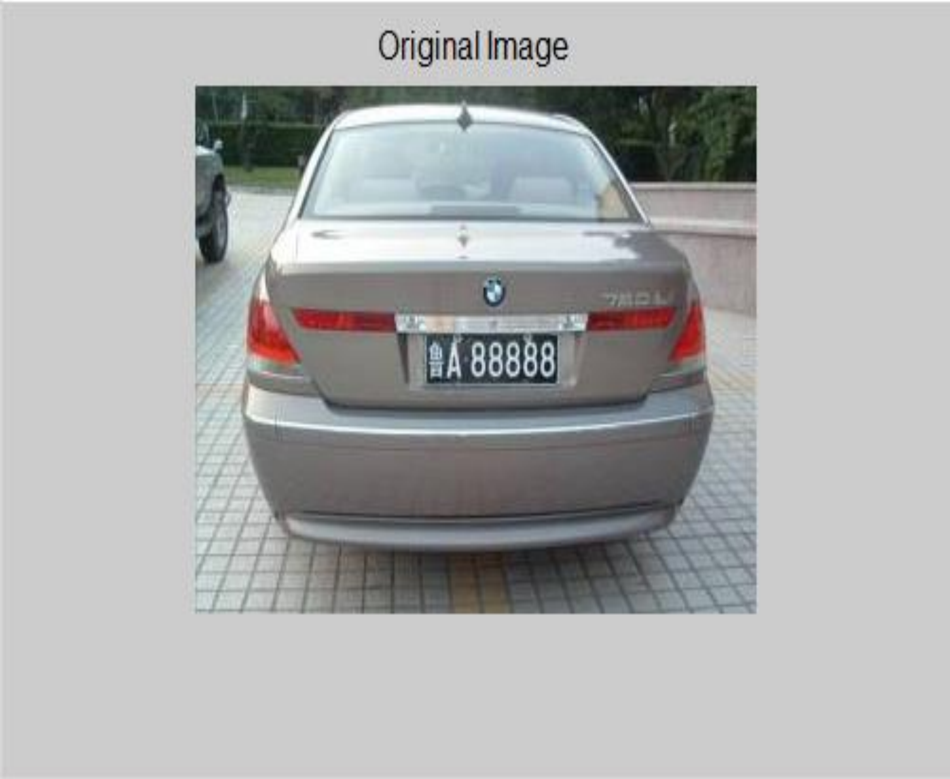

Figure 4.18. Whole car with the car license plate

In these cases, we test our proposed filter, and find that our method can still detect the car license plates by simply adjusting the threshold of the x-axis and y-axis. We adjust the threshold of x-axis to 0.7 and keep the threshold of y-axis unchanged (0.5). Then our method matches that kind of median size car license plates. The results of our experiment are shown in Figures 4.19, 4.20 and 4.21.

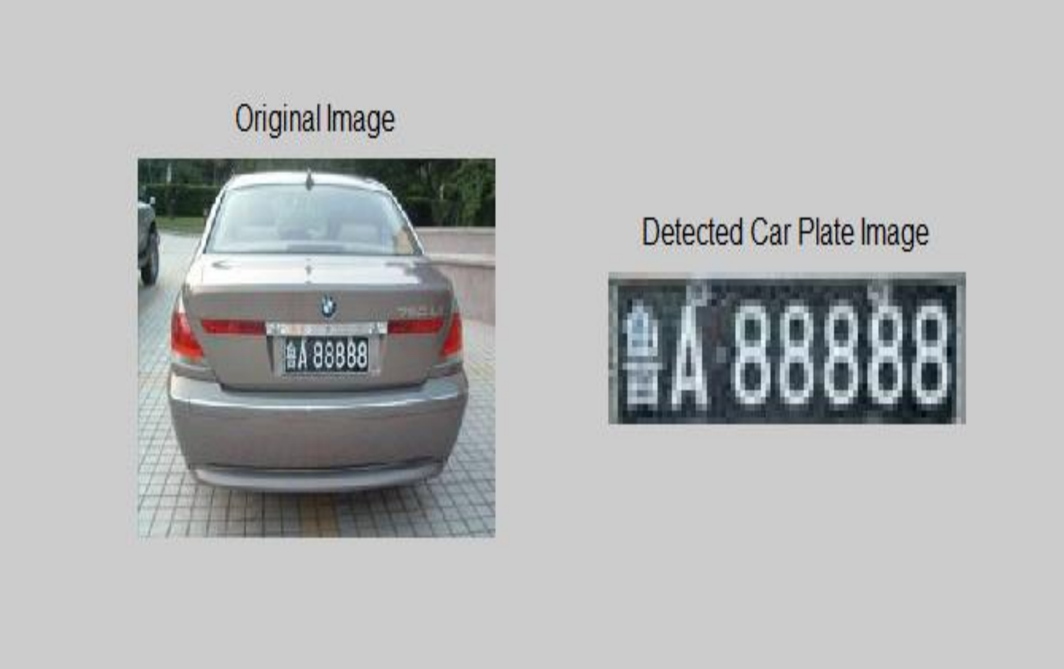

Figure 4.19. The detected car license plate-4

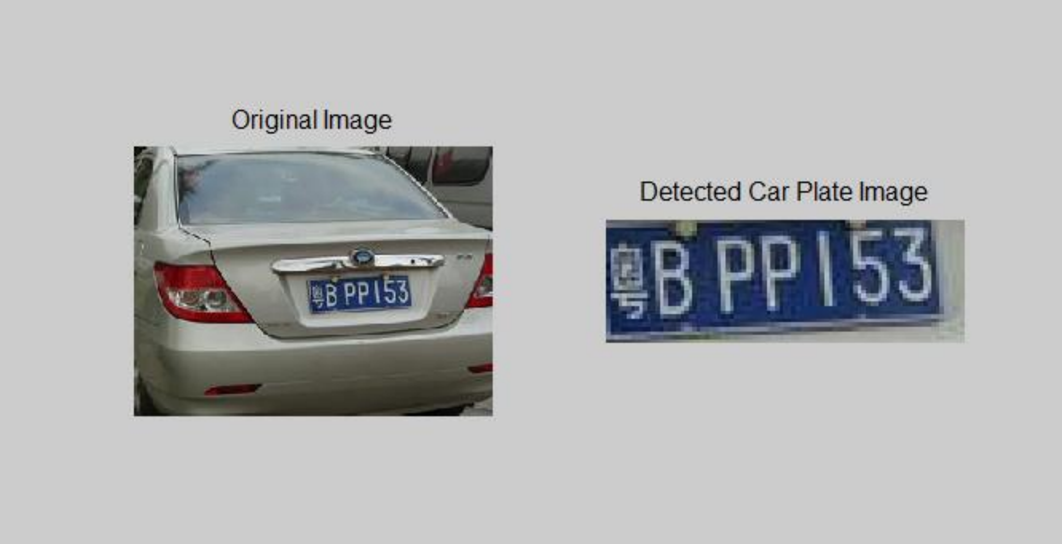

Figure 4.20. The detected car license plate-5

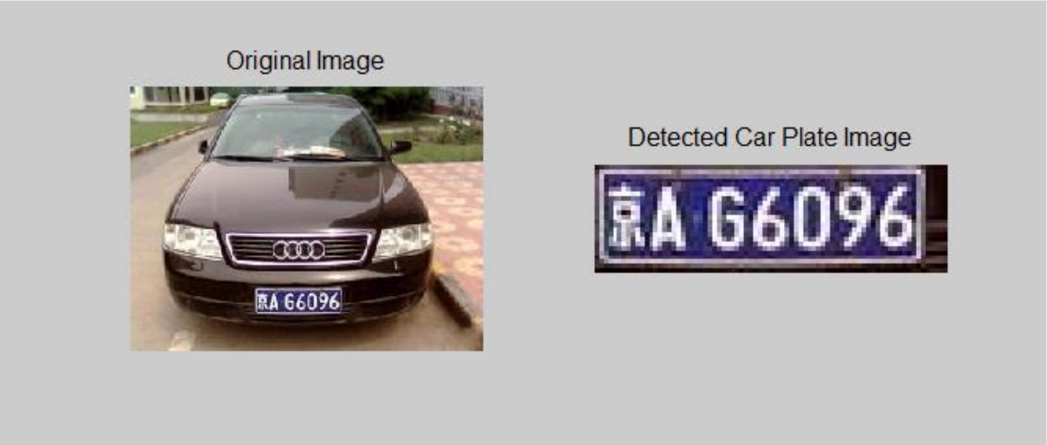

Figure 4.21. The detected car license plate-6

It is a simple method to pick out the highest energy frequency area of a vehicle image, in our cases which are the vehicle license plates. For other similar sized vehicle images, the parameter settings may be chosen as the same, hence it is useful for urban surveillance systems. For example, at a car parking lot, the camera takes the vehicle images at a fixed place and fixed distance with the vehicles.

4.4.2 Use Our Filter to Detect Vehicles

In the following, we will show that our unified filter can be used to detect the vehicles from the images, since the vehicles also show high energy frequency within the images.

This time we begin from a car image shown in Figure 4.22. Our aim is to pick out the car from the image. The first step is to convert the color image into a grey image. Then, similar to the car license plate searching, we calculate and obtain the gradient of the grey image.

After we complete the pre-treatment process of the car image, the next step is to pick out the car using our proposed method.

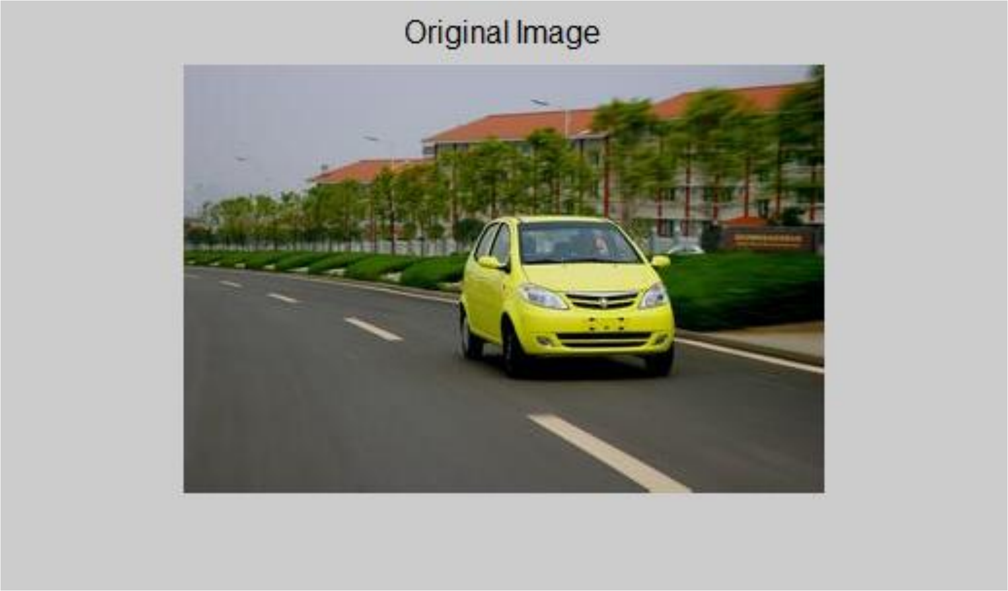

Figure 4.22. The original image of a car

We first filter the pre-treated gradient image in horizontal direction. In this direction, the high frequencies areas are shown out by brighter area. In order to get the intuitive impression, we plot the one-dimensional brightness values for straightforward view by choosing the middle horizontal array line of the filtered image. The array values are shown in Figure 4.23.

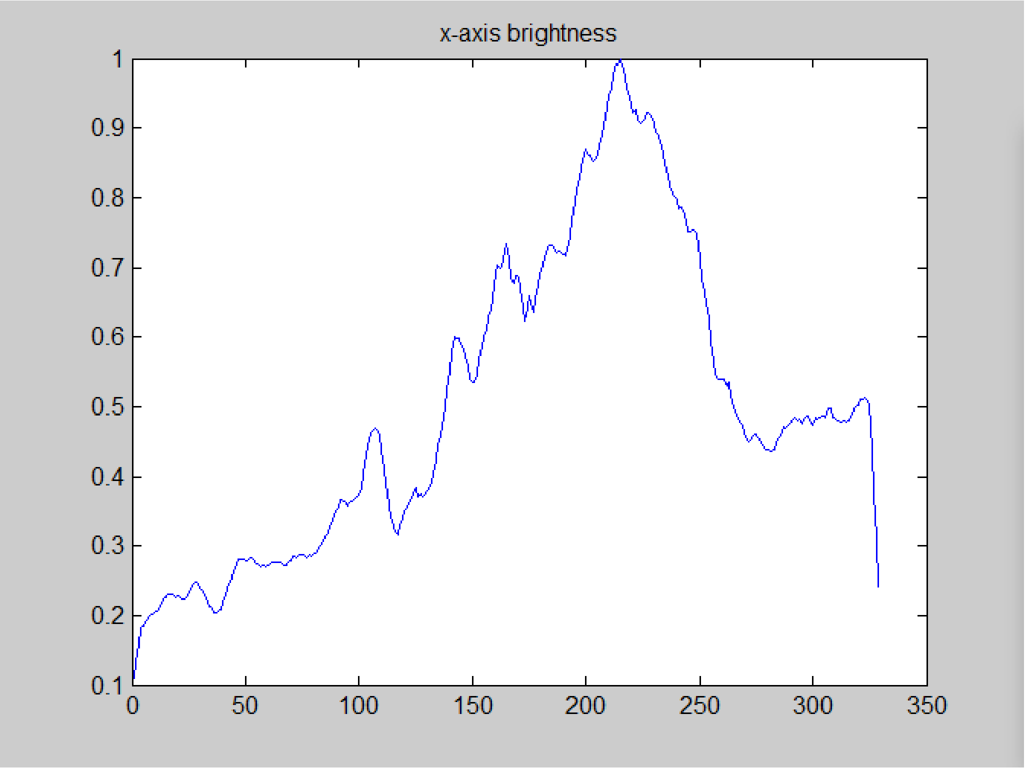

Figure 4.23. The horizontal axis's brightness values

Since the prepared work has been done perfectly, to cut out the car area is not difficult now. As for the horizontal axis, the car area is the brightest area. According to this feature, we only need to scan the horizontal axis array from left to right. We simply choose the threshold at 0.5 for the horizontal axis. In order to eliminate the interruption of unwanted brightness, we check the picked width between the thresholds. Since the width of the car is within an already-known area, we can cut out the car in horizontal direction correctly; the result is shown in Figure 4.24.

Similar to the horizontal direction, we use matrix array filter in the perpendicular position. Since the car area in horizontal direction has already been picked out, we only need to filter the pre-treated image in the area shown in Figure 4.24.

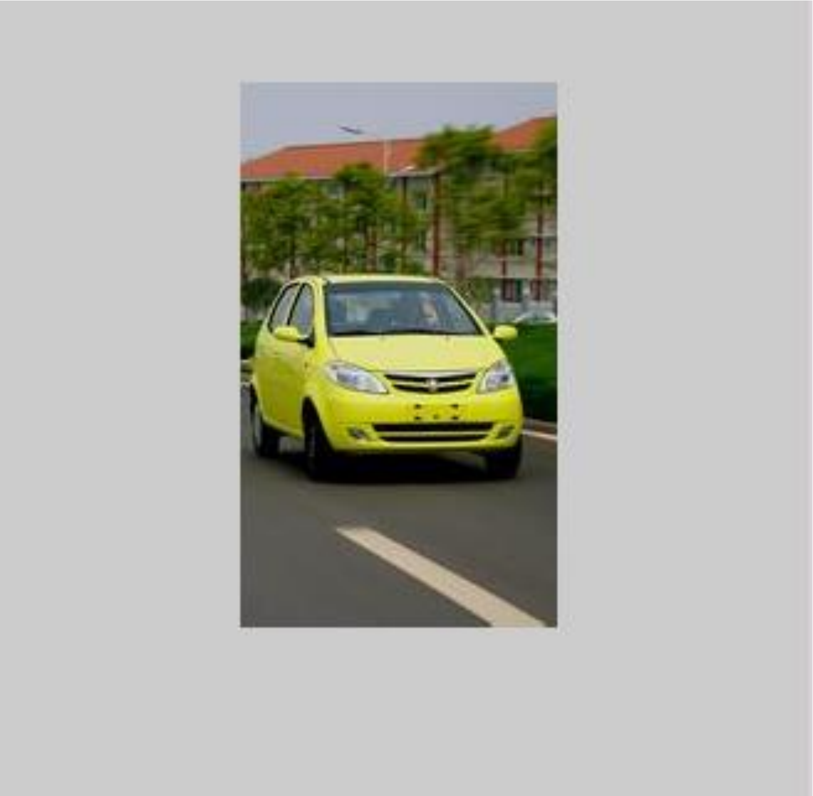

Figure 4.24. The detected car area in horizontal direction

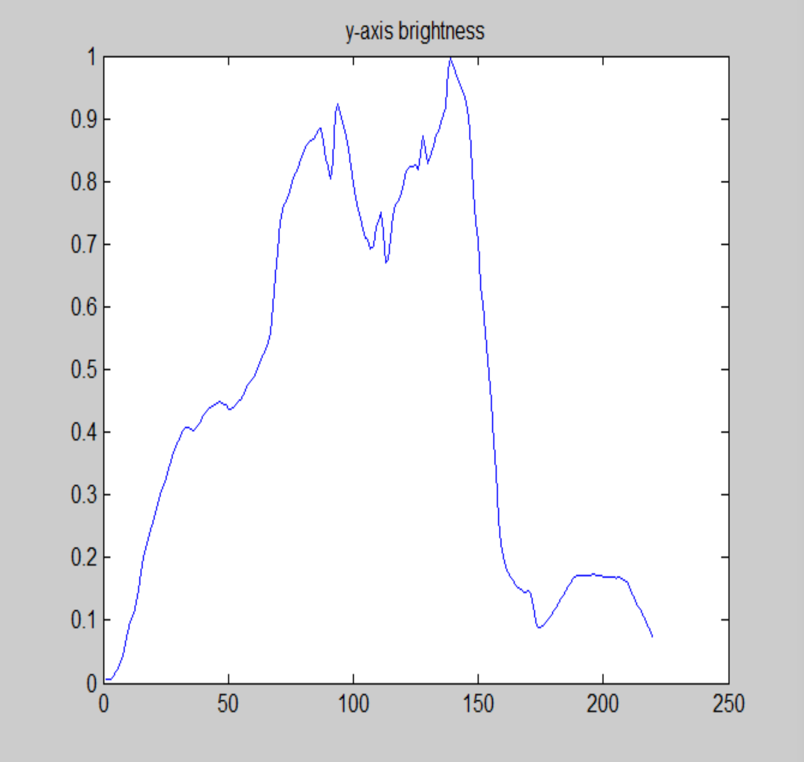

Figure 4.25. The perpendicular axis's brightness values

Now we are to cut out the car area in perpendicular direction. As for the perpendicular axis, the whole car area is the brightest area although there are some interruptions of noise. According to this feature, we only need to scan the perpendicular axis's brightness values from left to right and simply choose the threshold as 0.5 for the perpendicular brightness values. In the scan process, we check the picked height between the thresholds. Since the height of the car is within an already-known scale, we can cut out the car in perpendicular direction correctly.

After getting the four edge value, we cut out this area from the original car image, and the result is shown in Figure 4.26.

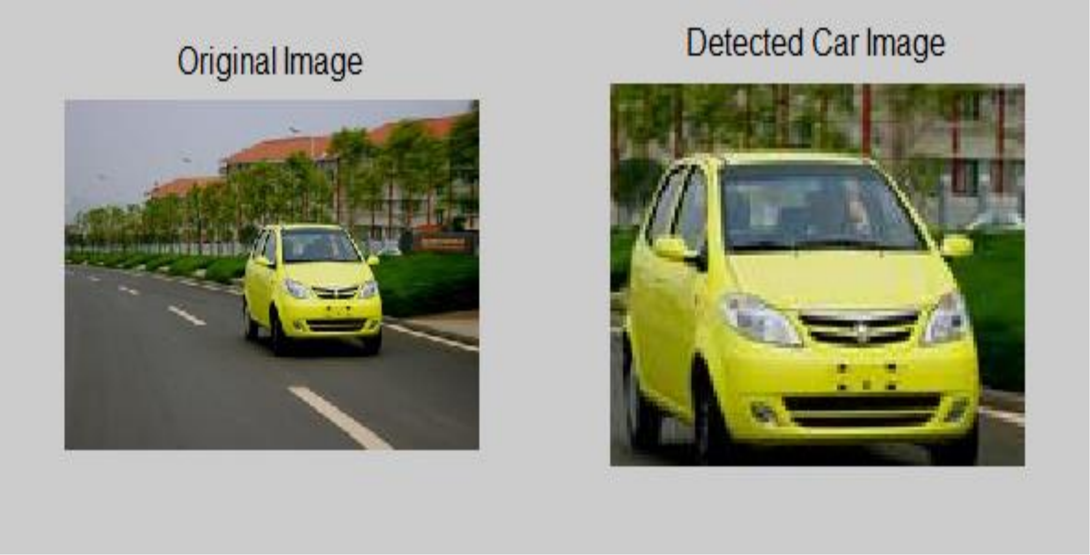

Figure 4.26. The detected car-1

The result is amazing. Note that we use the same simple and effective filter to get the result.

Then, we test our filter to see how it works on more car images. Similar results are shown in Figures 4.27 and 4.28 with the same software code. We find that our simple filter can filter out the cars from their images correctly. This filter is not only suitable for searching the vehicle license plates from the vehicle images, but also suitable for detecting the vehicles from different environments, which is very useful in the IoT applications of smart cities, since this filter can flexibly find out the vehicle information based on different demands of urban surveillance systems.

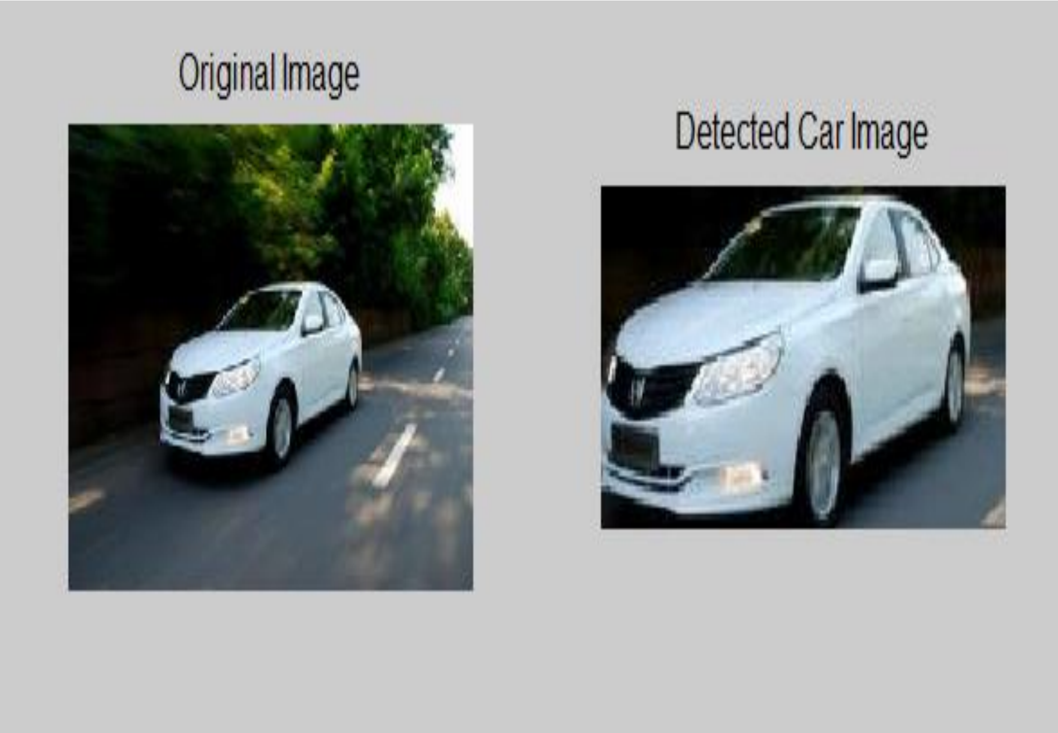

Figure 4.27. The detected car-2

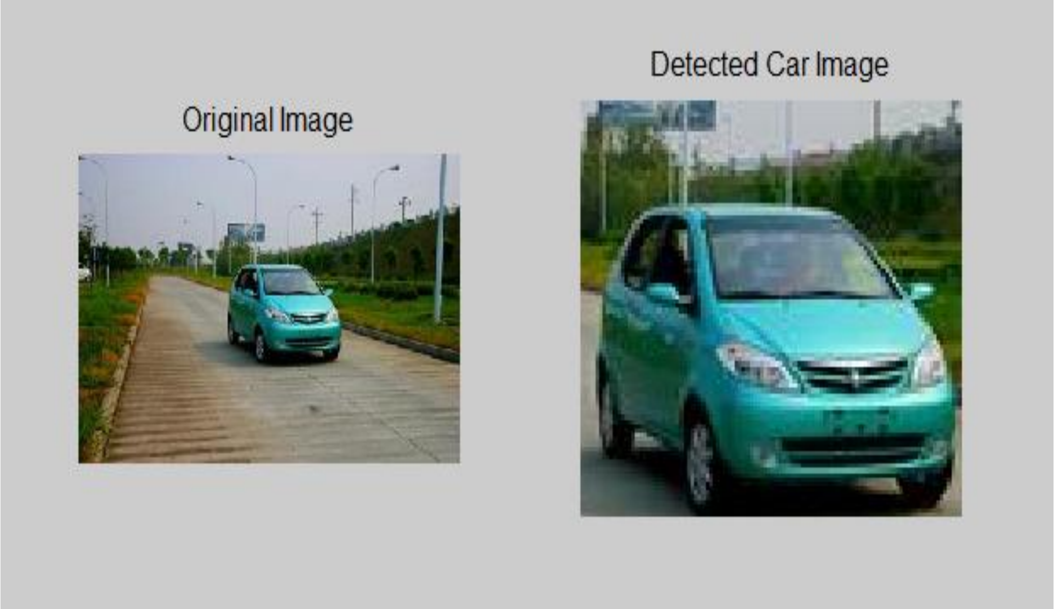

Figure 4.28. The detected car-3

4.5 Summary

This chapter designs a novel filter which is unified to detect either the vehicle license plates or the vehicles from the digital camera imaging sensors of urban surveillance systems in smart cities. To the best of our knowledge, we are the first to design this kind of filter to detect the vehicle/license plate objects. We tested our filter with different images. The results show that our filter can automatically detect the highest energy frequency areas out from the images, which makes our proposed algorithm a simple and effective method to automatically detect vehicle objects for IoT and smart cities applications. Our method can also be used to reduce the big data volume which is generated every day from urban surveillance systems in smart cities.

CHAPTER 5

BIG DATA ORIENTED NOVEL BACKGROUND SUBTRACTION ALGORITHM FOR URBAN SURVEILLANCE SYSTEMS

Part of this chapter was published as a research paper below:

L. Hu, Q. Ni, F. Yuan, “Big data oriented novel background subtraction algorithm for urban surveillance systems”, Big Data Mining and Analytics, Volume: 1, Issue: 2, Pages: 137-145, June 2018.

Due to the tremendous data generated by urban surveillance systems, big data oriented low-complexity automatic background subtraction techniques are demanded. In this chapter, we propose a novel method of automatic background subtraction algorithm for urban surveillance systems. In this way the computer can automatically renew the image as the new background when no object is detected. This method is simple and also robust to the impact of light changing.

5.1 Introduction

Big data research attracts great attention recently due to infinite generation of huge data worldwide. For example, big data oriented techniques emerge as important research topics in smart cities and urban surveillance systems [69][70].

Urban surveillance systems are important applications for smart cities [71][72]. Cameras are installed in these applications to automatically detect vehicles and other objects through automated object detection methods. Because of the tremendous data generated by urban surveillance systems, in order to get useful information from huge images and videos, low-complexity techniques which are able to automatically identify objects from various sources are highly demanded. Hence, automated or called automatic object detection algorithms become an important research topic in urban surveillance systems. Currently, the algorithms under investigation can be classified into three main groups, which are frame difference, background subtraction and optical flow calculation methods.

The frame difference algorithm is used to analyze the image sequences of two or more adjacent frames to identify the moving targets since the differences of these continuous frames are calculated. For every pixel, if the difference is larger than a settled threshold, the result will be set as “1”, else “0”. The larger the threshold value is chosen, the less noise is there [73][74]. In order to solve the problem of moving target detection and tracking, some authors propose the frame difference method to combine with particle filter algorithm [75]. The shortcoming of the frame difference method is: it is not sensible to detect unmoving object since there will be no difference between the adjacent frames.

Another type of method is background subtraction method. This method firstly builds the scene background model, and then uses the current frame image to subtract the background model. In this way the targets are picked out. It is a widely used method to fetch targeted objects. The background subtraction performance depends mainly on the background modelling techniques, in which Gaussian mixture models (GMMs) are now the popular methods. In [76], the authors mix GMM with classified foreground in an interpolated red, green, and blue (RGB) domain. Since one problem of GMM is that it cannot model noisy backgrounds properly, the researchers in [9] utilize advanced distance measure based on support weights and histogram of gradients for background suppression to remedy such a shortcoming of GMM. Furthermore, current background subtraction methods (such as GMM) are limited by their shortcoming of high computational complexity [77][78], e.g. the large number of calculations limit their applications in real-time because the reconstructions and updates of the backgrounds take long time.

The third method is based on optical flow (OF) calculation [79]. It needs to estimate the motion scenes and combine similar motion vectors to detect moving objects. However, the OF method is very computationally expensive. Another main shortcoming is that OF method assumes the object is always moving, which is not always true in reality.

In this chapter, we propose a novel low-complexity automatic background subtraction method. Our method is computationally efficient and does not require that the objects are always moving.

5.2 Our Proposed Algorithm

Our idea is to detect whether or not there are objects (e.g. cars, humans) inside each image. If there is no object detected, we will update the current image to be the background. In this way the background will be renewed in real time. We know that sometimes the sunlight background changes fast, but our methos is robust to the impact of light changing. Compared to existing methods which are computationally expensive, our method is very computationally efficient. Another advantage of our method is that it does not depend on the moving situation of objects; even the unmoving objects will be detected automatically as well. In [80], we proposed a simple filter to detect vehicles or license plates from the captured images. Our finding is that an object (for example a vehicle, human being) is normally the highest energy frequency part of an image and the energy frequency curves reduce sharply beyond the object boundaries. In this chapter we will improve the design of filter to detect and update the background. Our proposed filter is able to determine automatically whether there are objects inside the images.

5.2.1 Pre-Treatment of the Captured Image

Let us take an example as shown in Figure 5.1. Our objective is to let a computer to automatically determine whether there are objects on a road in this image captured. In this way the computer can determine whether or not this image can be used as a background. Using the pre-treatment method proposed in [80], we firstly transfer the original image into grey scale image. It reduces the image data needed to be calculated.

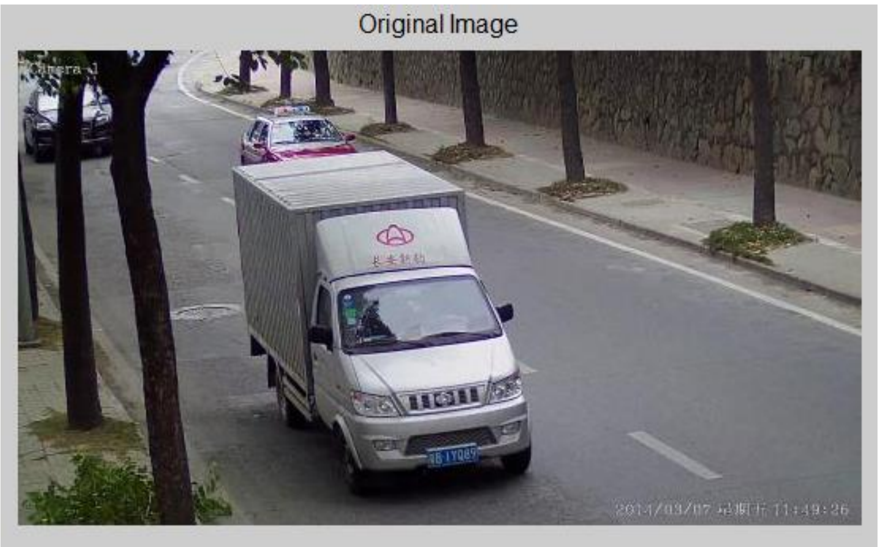

Figure 5.1. The original image

After the grey image is prepared, we calculate its gradient. We use P_x to denote the differences in x direction (horizontal) and P_y to denote the differences in y direction (vertical). As in [80], after calculating the gradients of both the horizontal and vertical directions, for every pixel, we can obtain the overall gradient of this pixel as follows:

$$P = \sqrt{{P_x}^2 + {P_y}^2}\,. \qquad (5.1)$$

The gradient of this grey image is illustrated in Figure 5.2, where some large areas of constant grey scale fade into darkness due to the slow movement of gradient. Only the outline of the vehicles and some other background are highlighted.

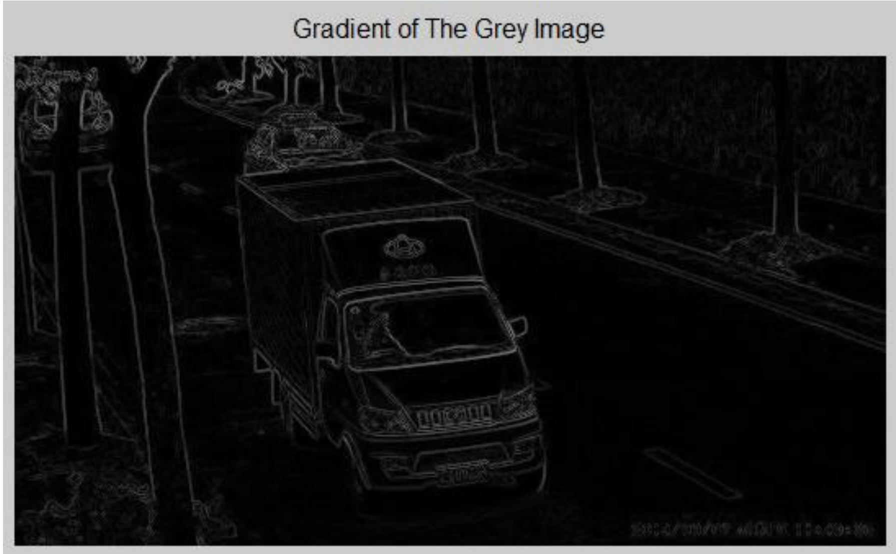

Figure 5.2. The gradient of grey image

5.2.2 Design of Our Effective Filter

In [11], we designed a smart filter based algorithm to scan and pick out the object area of the image. Our method was motivated by the characteristic of mathematical definition of intercorrelation and the function pulse δ(x), which holds an important characteristic as follows:

$$\int_{-\infty}^{\infty} \delta(x-a)\varphi(x)dx = \varphi(a). \qquad (5.2)$$

This means that the function δ(x-a) can be utilized to filter out another function φ(x) at the x-axis when x=a. We use this property to filter out the high energy frequency area of an image.

We use matrix array to calculate the intercorrelation with an image. Since the matrix array is centrosymmetric, the intercorrelation process with an image can be considered as a convolution process, which is shown as follows:

$$\mathrm{F(n)} = \mathrm{G(n)} * \mathrm{h(n)} = \sum_{i=-\infty}^{+\infty} G(i)h(n-i), \qquad (5.3)$$

where G(n) presents the original image and h(n) represents the matrix array. $F(n)$ is the image after the convolution. As we know, the results obtained after the convolution reflect the overlapping relationship between the two convolution functions. In mathematics, convolution can be regarded as a weighted summation.

In the filtered image$F(n)$, since spectrum information is displayed on the horizontal axis in the form of brightness, by simply choosing the middle horizontal array line of the filtered image, we obtain the one-dimensional brightness values. These values reflect the high frequency areas (coming from the objects) on horizontal axis clearly. Our proposed filer successfully reduces the two dimensional image into one dimensional curve, which significantly reduces the data needed to be dealt with.

Since the matrix filter proposed in [80] is a bar shape, we call it a bar filter in this chapter. As we know, inside the bar filter, all the numbers are set as 1's. It can be considered as the constitution of some line arrays.

The bar filter incur a lot of calculations. In order to reduce the computational load but still get satisfactory convolution results, in this chapter we propose a new and improved filter. We remove some line arrays from the original matrix array. In this way, the new filter we are proposing looks like a grate; hence we call the proposed filter the grate filter. The diagrammatic sketch is shown in Figure 5.3.

In our cases, we set the width of the bar filter as 8, which means there are 8 columns of 1's to integrate this filter. For the grate filter, we set the first and the fourth columns as 1's, which means there are three columns of 0's between the 1's columns and there are totally only two columns of 1's left. From the definition of the convolution, it is easy to know that in this case the computational load of the grate filter is only a quarter of the original bar filter. It is clear that the first advantage for the grate filter is its significant reduction of the computational load.

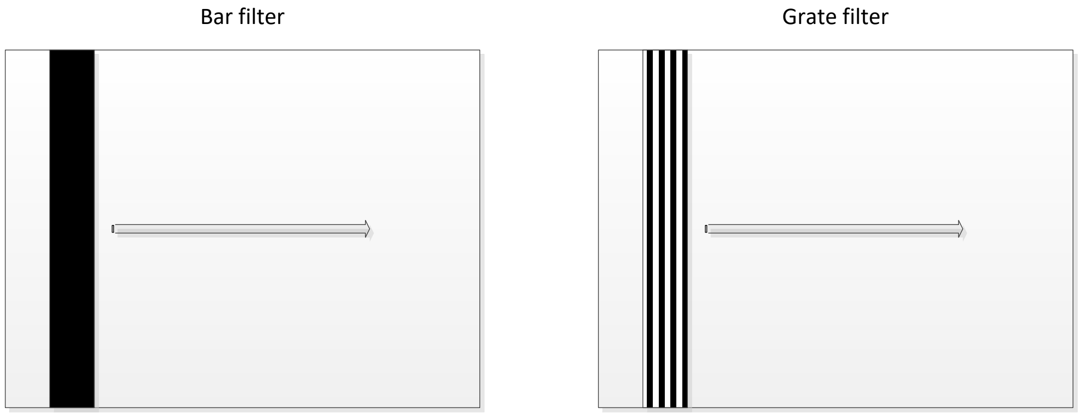

Figure 5.3. The bar filter (left) and our proposed grate filter (right)

We filter the pre-treated gradient image in the horizontal direction with the two different filters, and then plot the one-dimensional brightness values for straightforward view by choosing the middle horizontal array line of the filtered image. The values obtained in the x-axis are shown in Figure 5.4 (with bar filter) and Figure 5.5 (with grate filter).

We only need to scan the horizontal brightness values from left to right and simply choose the threshold at 0.5 for the horizontal axis. In order to eliminate the impact of unwanted brightness, we check the widths between the horizontal points with the value of the threshold and discard those very narrow results. If at least one of the detected widths is larger than the predefined width, the system indicates that there is at least one object in this image. In this case, since at least one object (i.e. a vehicle) is detected, the computer can reach the decision: “Object Detected”, which means the original image cannot be used to renew the background.

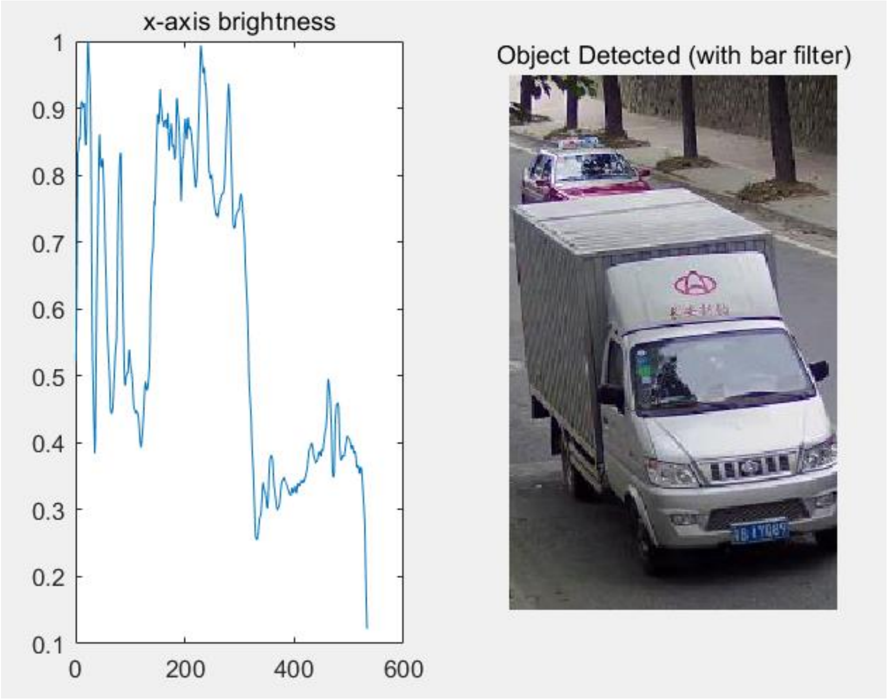

Figure 5.4. The horizontal axis's brightness values and picked result (with bar filter)

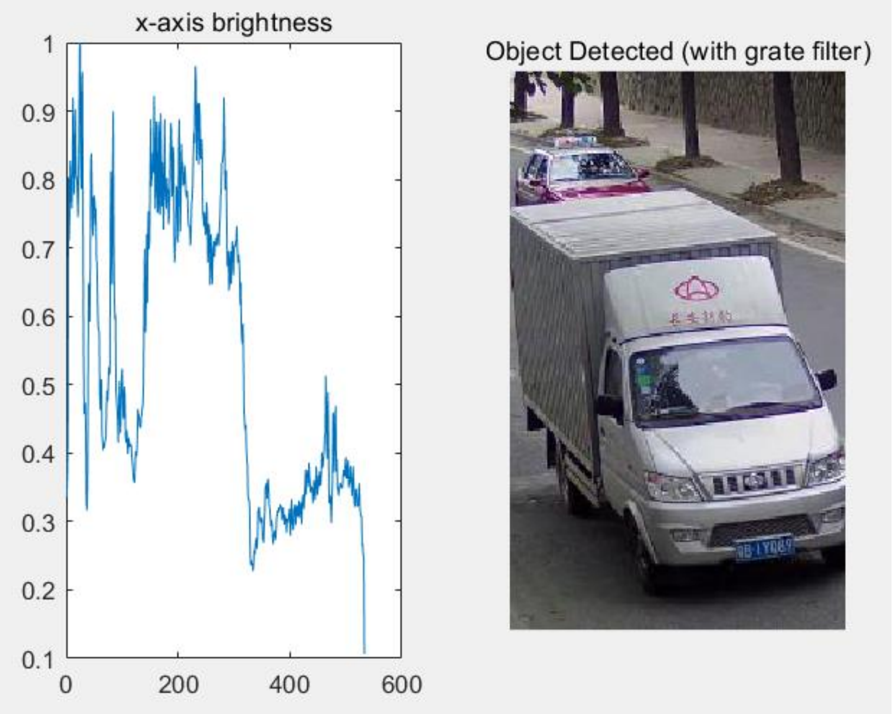

Figure 5.5. The horizontal axis's brightness values and picked result (with grate filter)

Figures 5.4 and 5.5 both show that at least one object is detected. Considering the reduction of the computational complexity, the grate filer outperforms the bar filter.

Moreover, the grate filter obtains more details in the horizontal axis's brightness values.

5.3 The Flow Chart of the Proposed Algorithm

The flow chart of our proposed method is illustrated in Figure 5.6.

The time period for taking a new image may range from several seconds to several minutes, the setting may be dependent on real situation of every surveillance area.

In the process of searching whether there are objects in an image, sometimes we can detect several different objects. Because our aim is to decide whether or not this image can be used to renew the background, it is not necessary to show all the objects inside a result image. We choose to show the first object detected from the image using our left to right rule. Under this situation, the former background image will be kept unchanged, and after the settled time period, a new image will be taking into consideration.

On the other hand, when no object is detected from the image, we will choose this image to update the background image, and our programme will show the original image with the title "Object not detected". After that, the same action will be repeated, i.e., after the settled time period, a new image will be taking into consideration.

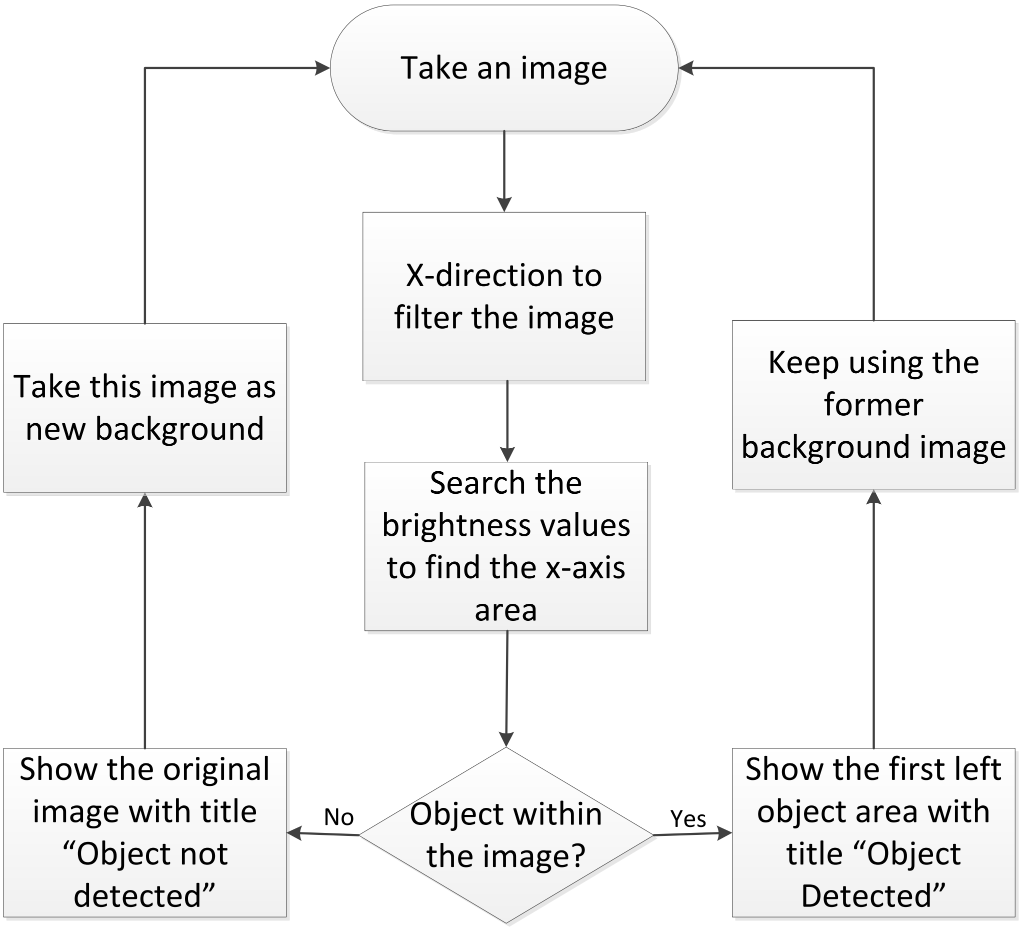
Take an image
X-direction to filter the image
Search the brightness values to find the x-axis area
Object within the image?
No
Yes
Show the original image with title “Object not detected”
Take this image as new background
Show the first left object area with title “Object Detected”
Keep using the former background image

Figure 5.6. The process of our proposed background subtraction method

5.4 Experimental Results of Our Proposed Filters

In this section, we test our filters to see how they behave on different images. We use various images captured on different roads to test the two filters. The original images and the identified results are shown in Figures 5.7 to 5.15.

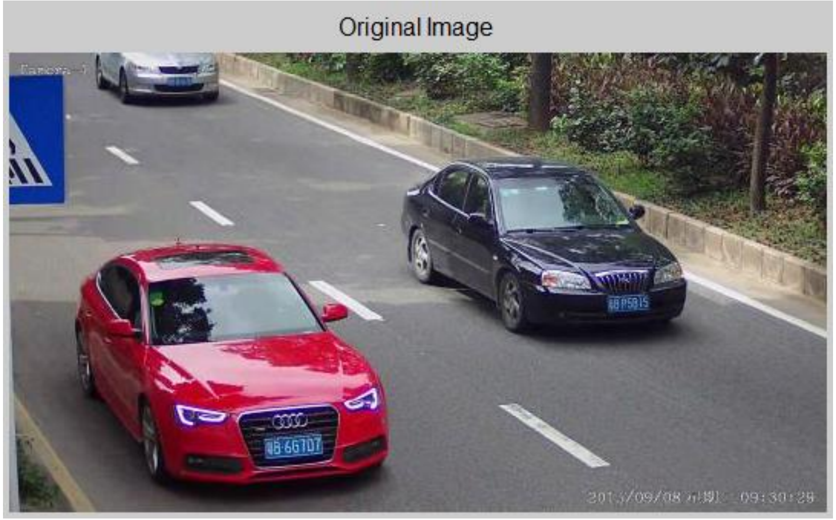

Figure 5.7. Cars on the road

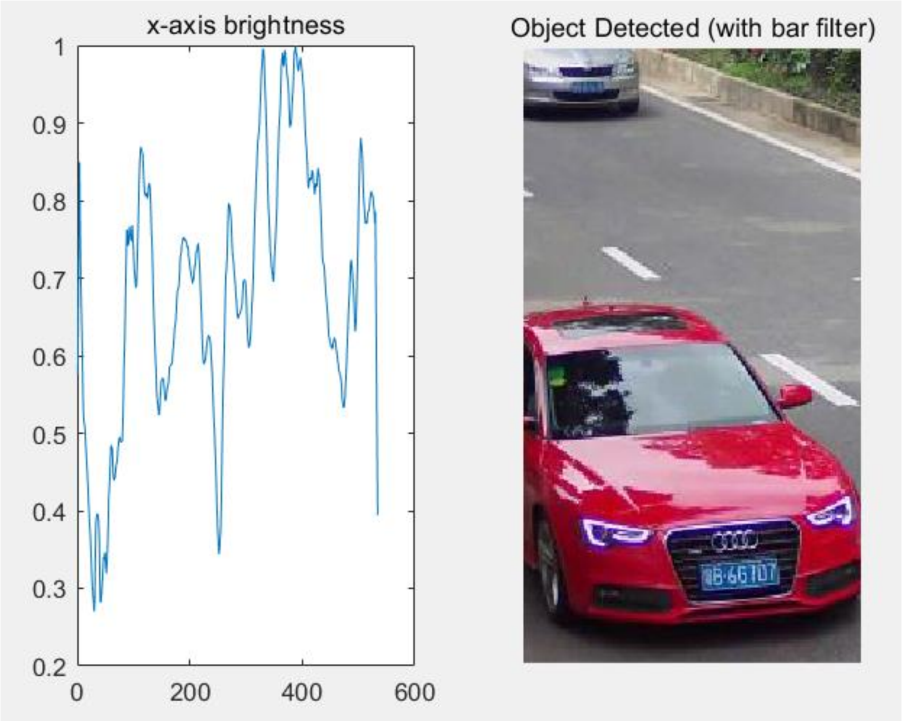

Figure 5.8. The horizontal axis’s brightness values and picked car (with bar filter)

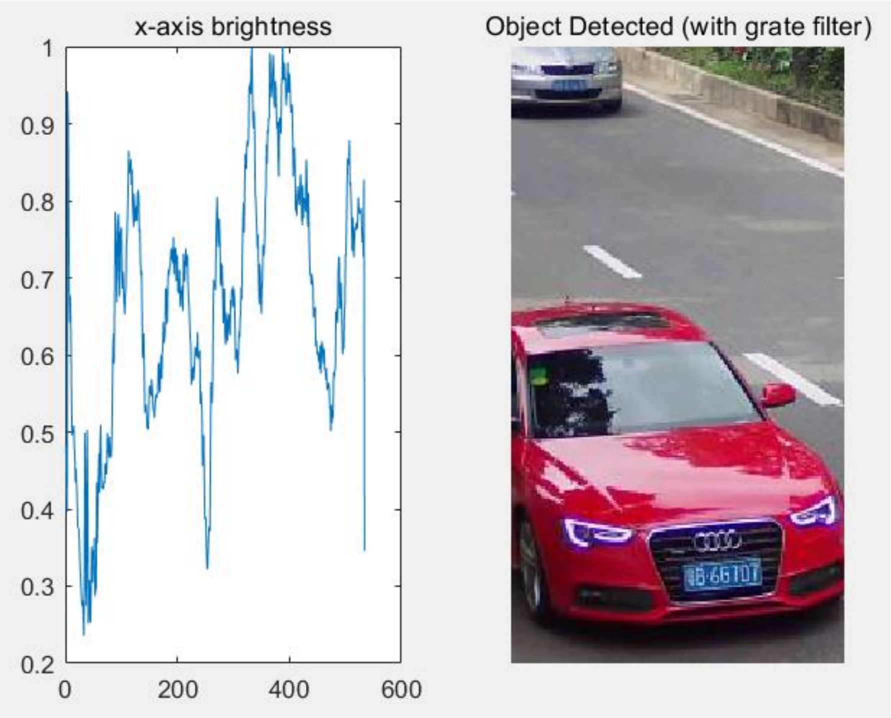

Figure 5.9. The horizontal axis's brightness values and picked car (with grate filter)

There are several cars in Figure 5.7. The horizontal axis's brightness values (Figures 5.8 and 5.9) clearly show several high frequency areas, which indicate there is at least one object in the image. The two filters both identify these cars and illustrate the first detected red car. The difference is that the grate filter shows more details in the horizontal axis's brightness values. Both images get the result "Object Detected". In this case the original image cannot be used as a background image.

There are more vehicles (buses and cars) inside Figure 5.10. The horizontal axis's brightness values (Figures 5.11 and 5.12) indicate that there are several high frequency areas which conclude that there are some objects in this image. Both filters detect these vehicles properly. The grate filter shows more details than the bar filter in the horizontal axis's brightness values again. It is clear that in this case the original image cannot be used as a background image.

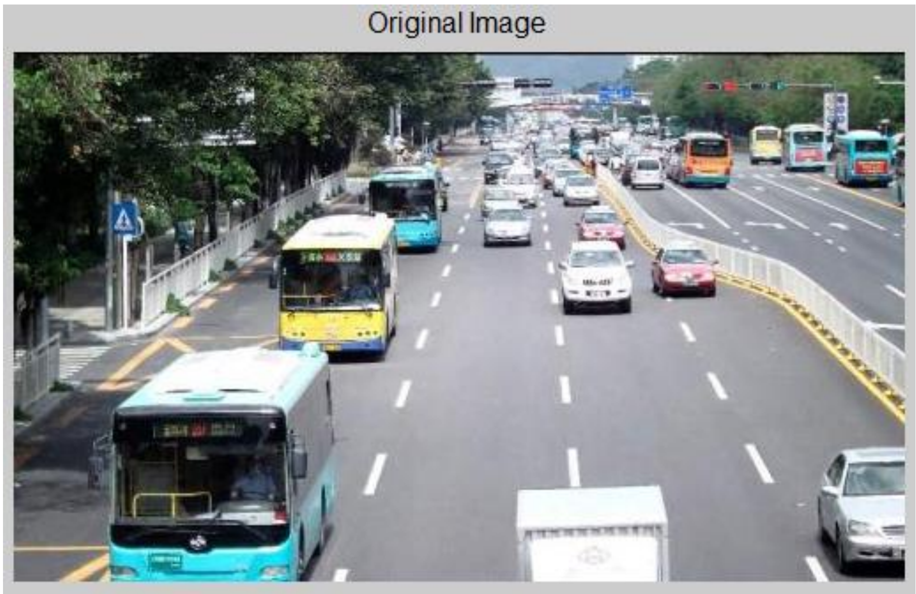

Figure 5.10. Vehicles on the road

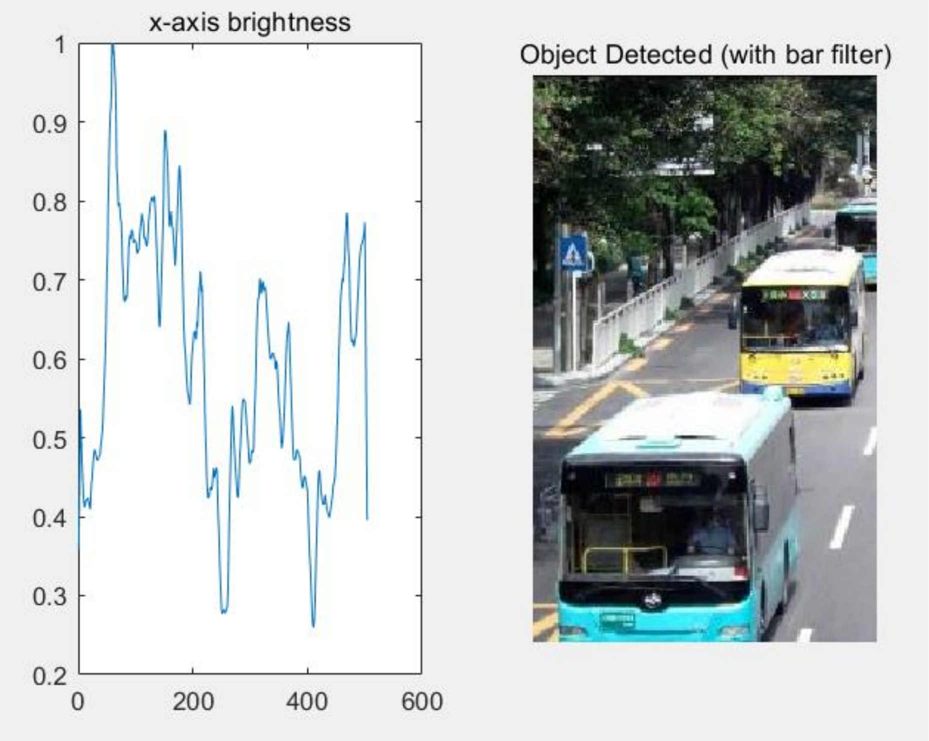

Figure 5.11. The horizontal axis's brightness values and picked bus (with bar filter)

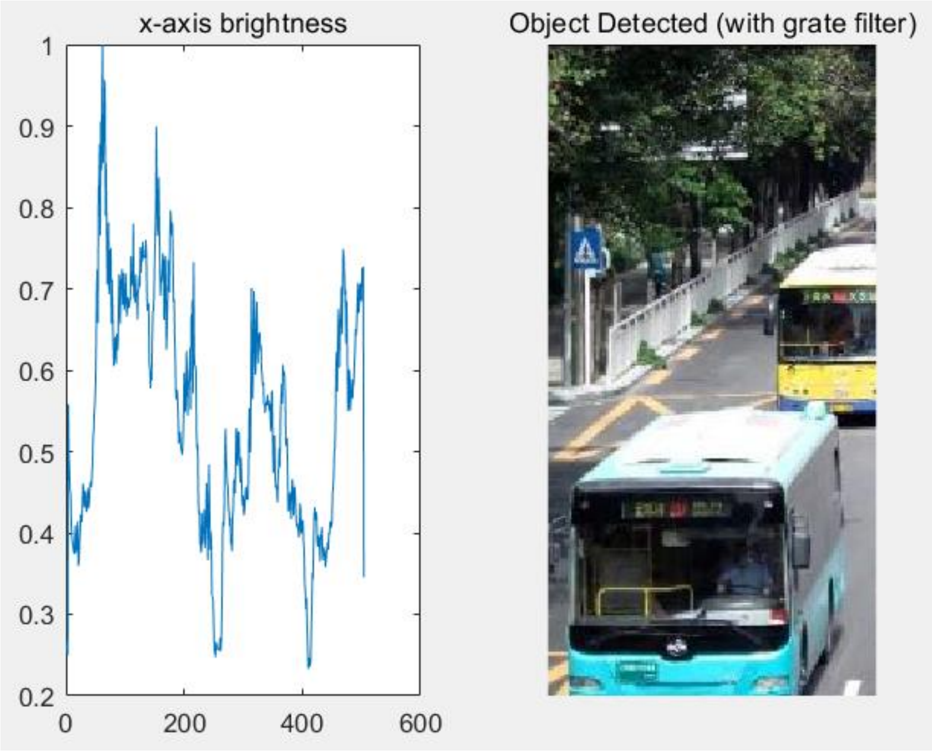

Figure 5.12. The horizontal axis's brightness values and picked bus (with grate filter)

In Figure 5.13 there are mainly two cars which are very close to each other. The horizontal axis's brightness values (Figures 5.14 and 5.15) show two high frequency areas, but the two areas are very close to each other. Although both filters can detect that there are objects on the road, the different filters achieve different precision. Since there are some smooth effects coming from the bar filter, the two cars are detected but mixed together in Figure 5.14. On the other hand, the grate filter outperforms the bar filter in detecting the details. The two cars are divided, and the first car is picked out by the grate filter clearly as shown in Figure 5.15.

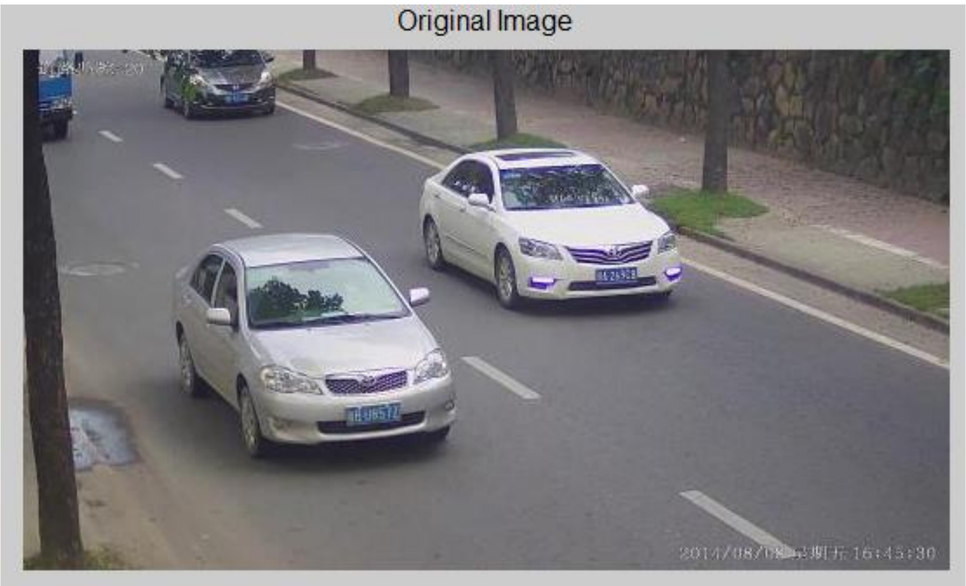

Figure 5.13. Cars on the road

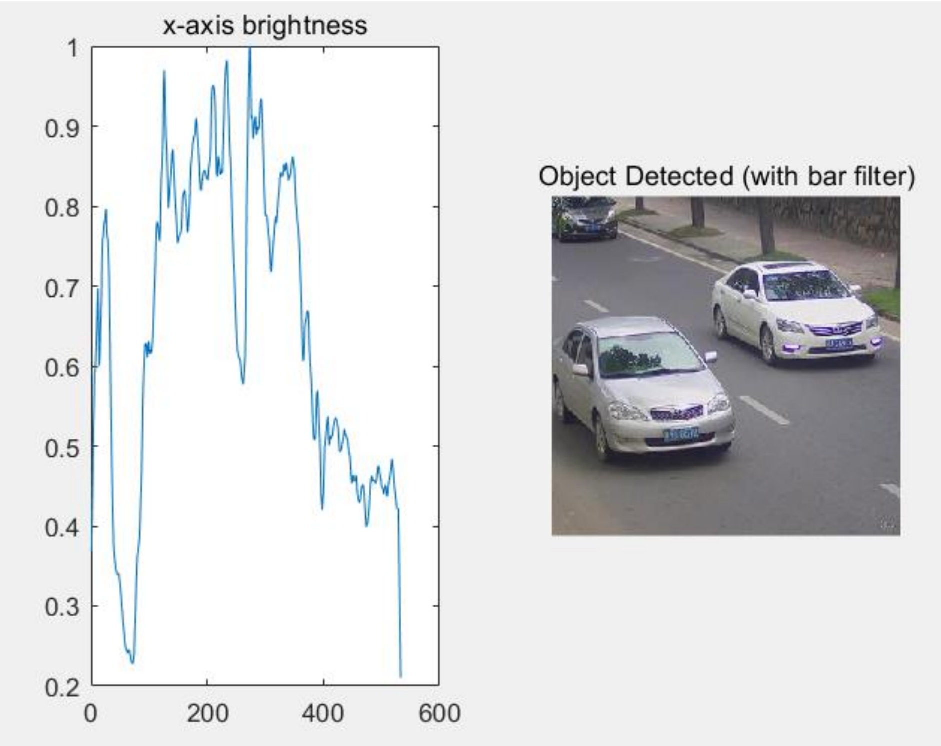

Figure 5.14. The horizontal axis's brightness values and picked cars (with bar filter)

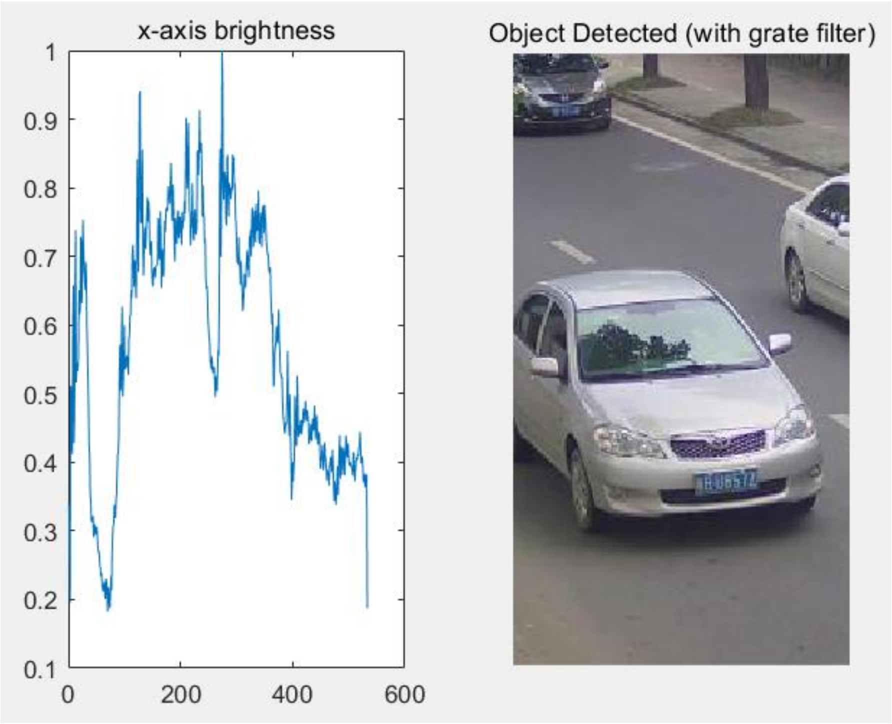

Figure 5.15. The horizontal axis's brightness values and picked cars (with grate filter)

Now we use some images of empty road to test our proposed algorithm. The original images and the detected results are shown in Figures 5.16 to 5.27.

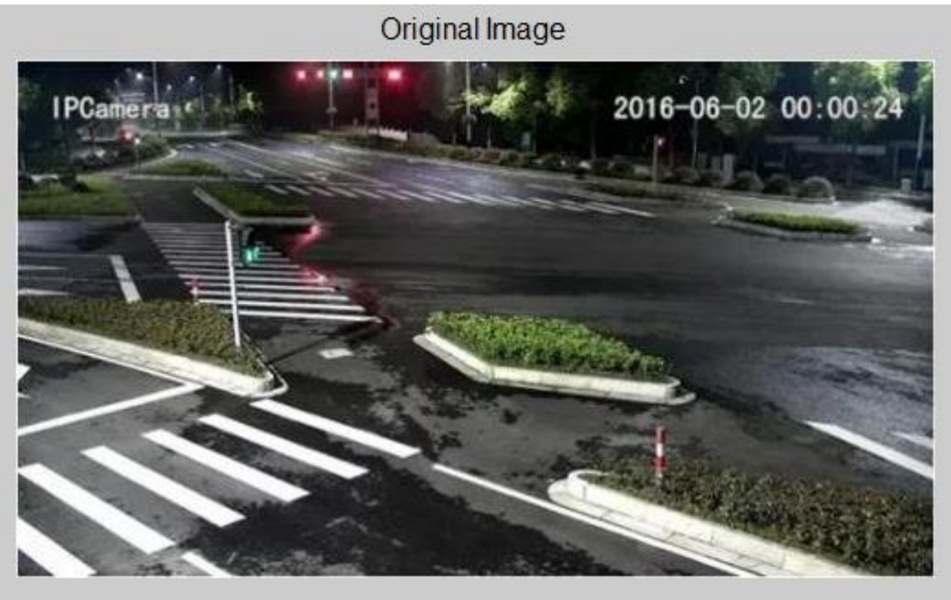

Figure 5.16. Empty road (1)

In Figure 5.16, neither vehicles nor human beings are on the road. However, there are several flower beds and lamp posts in the image which may affect our search process.

Figures 5.17 and 5.18 demonstrate that the flower beds and lamp posts cannot form wide high frequency areas. The high impulse at the right end of each figure comes from the edge of the original image. Although all the noise cannot meet our threshold demand, there is still some difference between the two filters. With the grate filter, the noise values are lower, which even cannot reach 0.5, that is, a better result. At last, both the results show as the title: “Object not detected”. This image is properly recognized to be the background image.

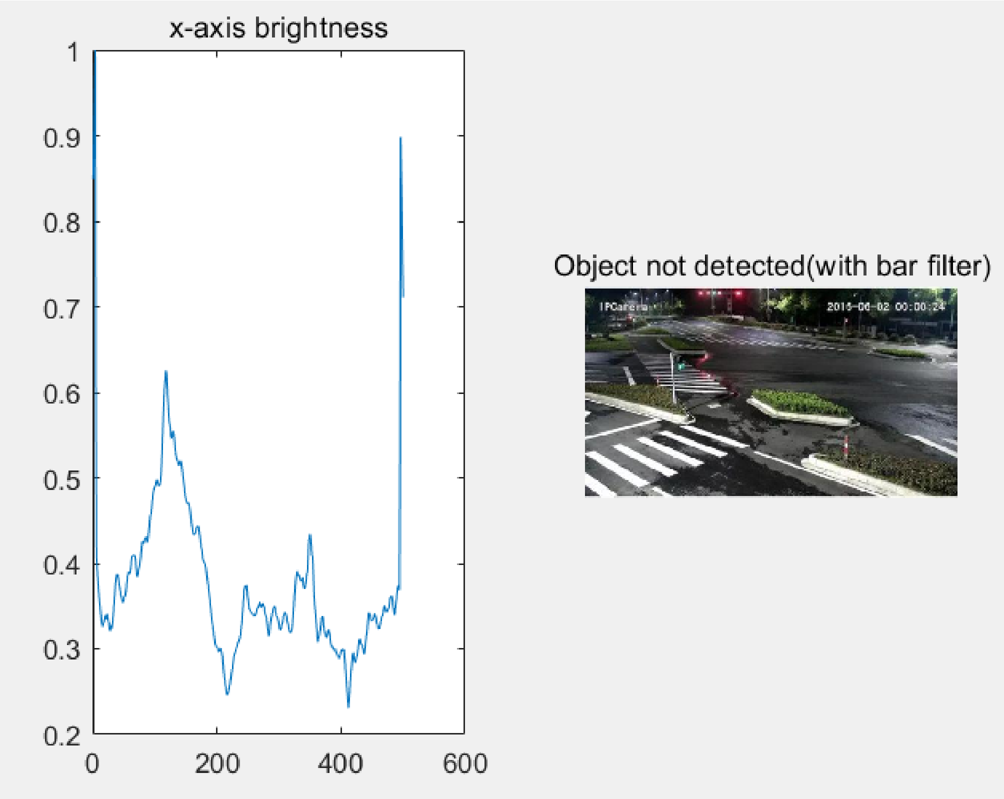

Figure 5.17. The horizontal axis’s brightness values and searched result (with bar filter)

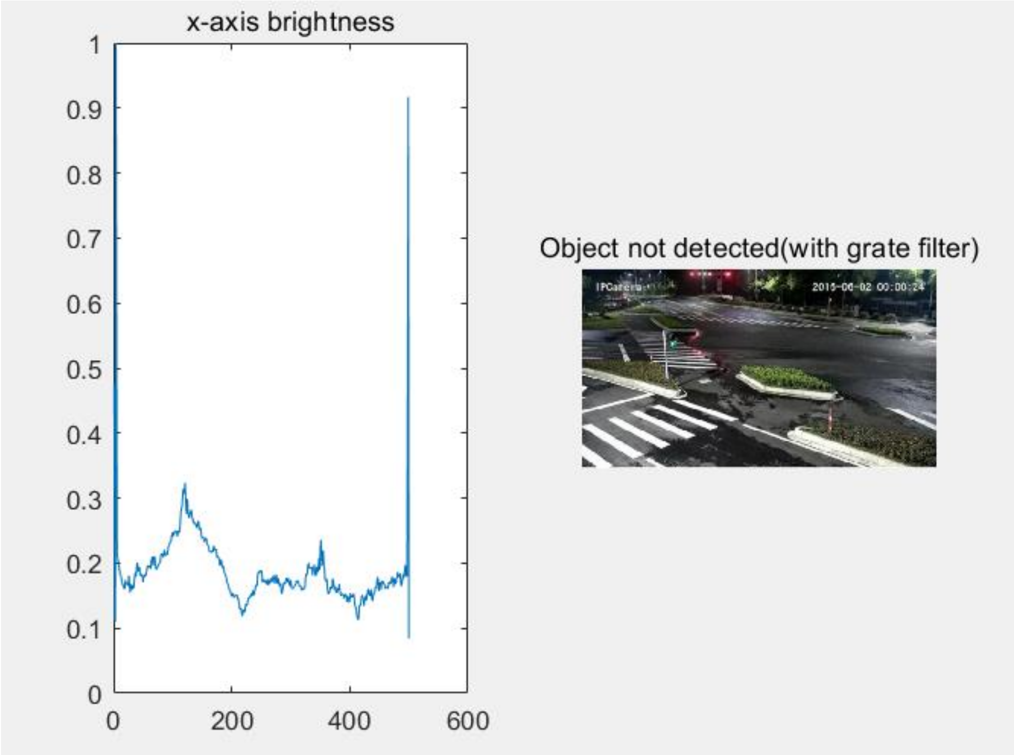

Figure 5.18. The horizontal axis’s brightness values and searched result (with grate filter)

Figure 5.19 is another image of empty road. The background contains some trees, buildings, and several lamp posts. Figures 5.20 and 5.21 show that they only form some narrow high frequency areas, and again the high impulse at the right end in each figure comes from the edge of the original image. Although the grate filter gives more details on the result brightness values, all these noise cannot meet our threshold demand, so they will not be considered as object. At last, both the filters conclude: “Object not detected”. This properly indicates that the original image can be used to update the background.

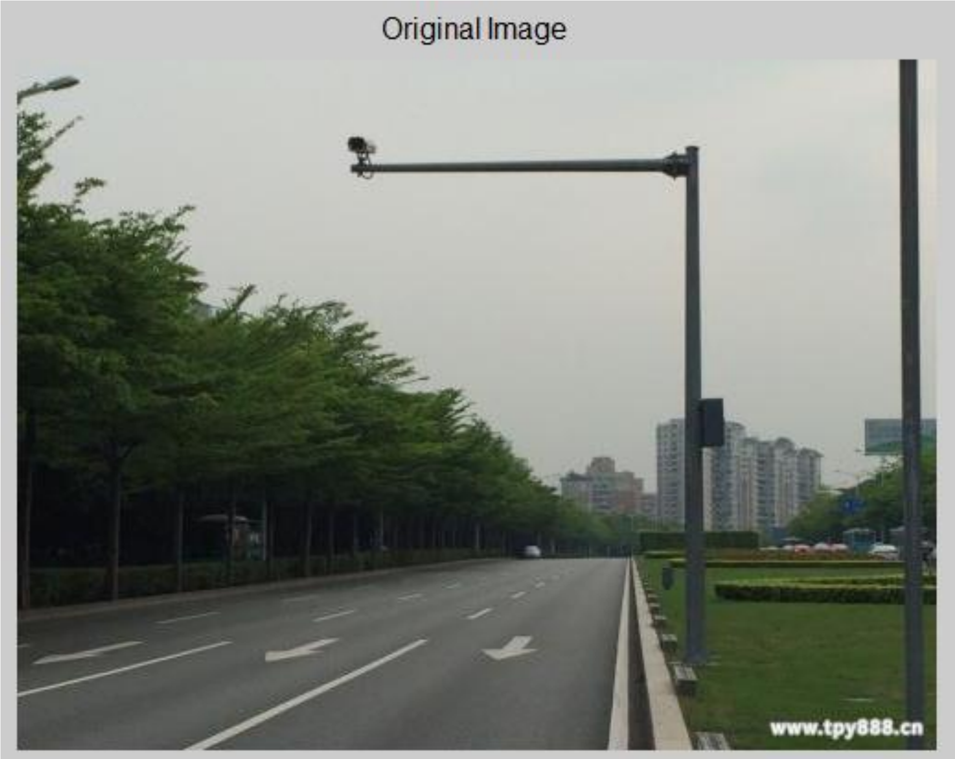

Figure 5.19. Empty road (2)

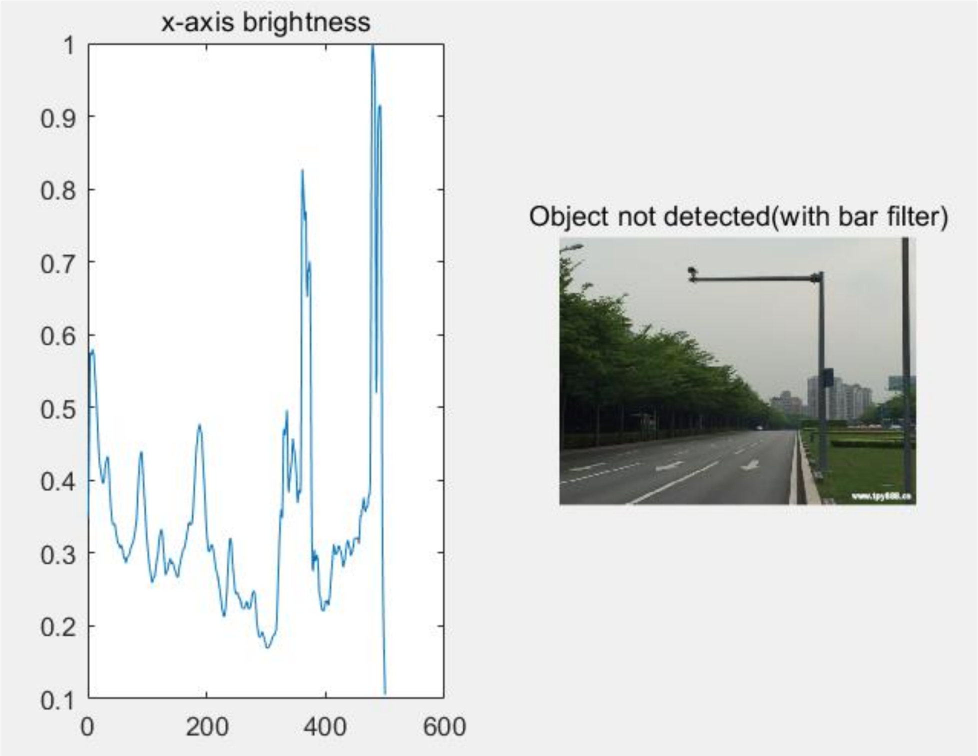

Figure 5.20. The horizontal axis's brightness values and searched result (with bar filter)

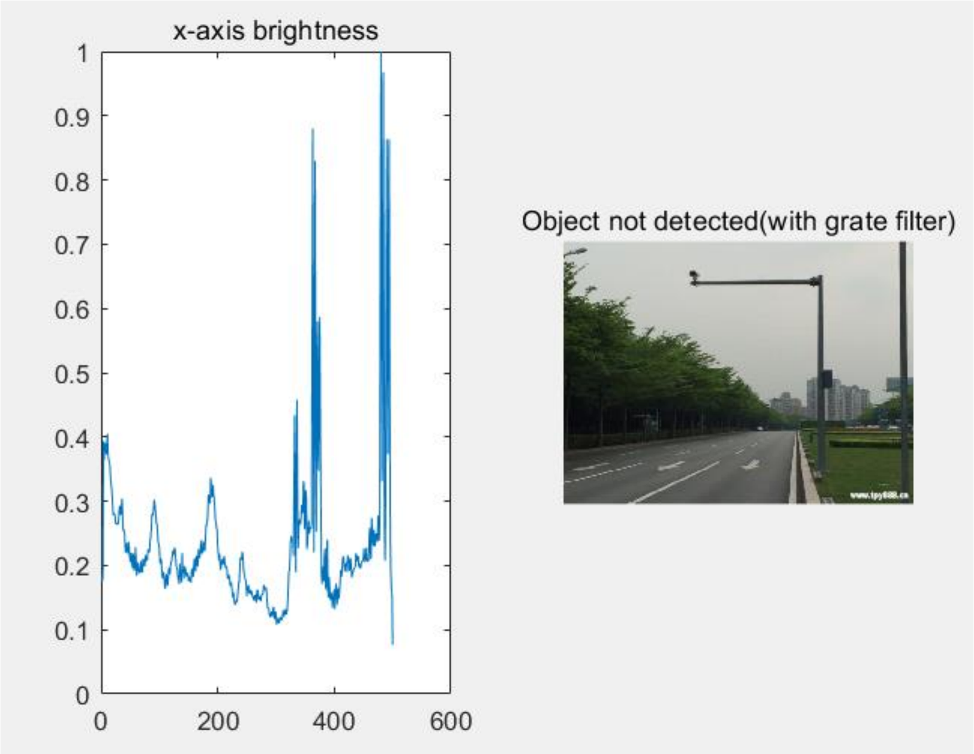

Figure 5.21. The horizontal axis's brightness values and searched result (with grate filter)

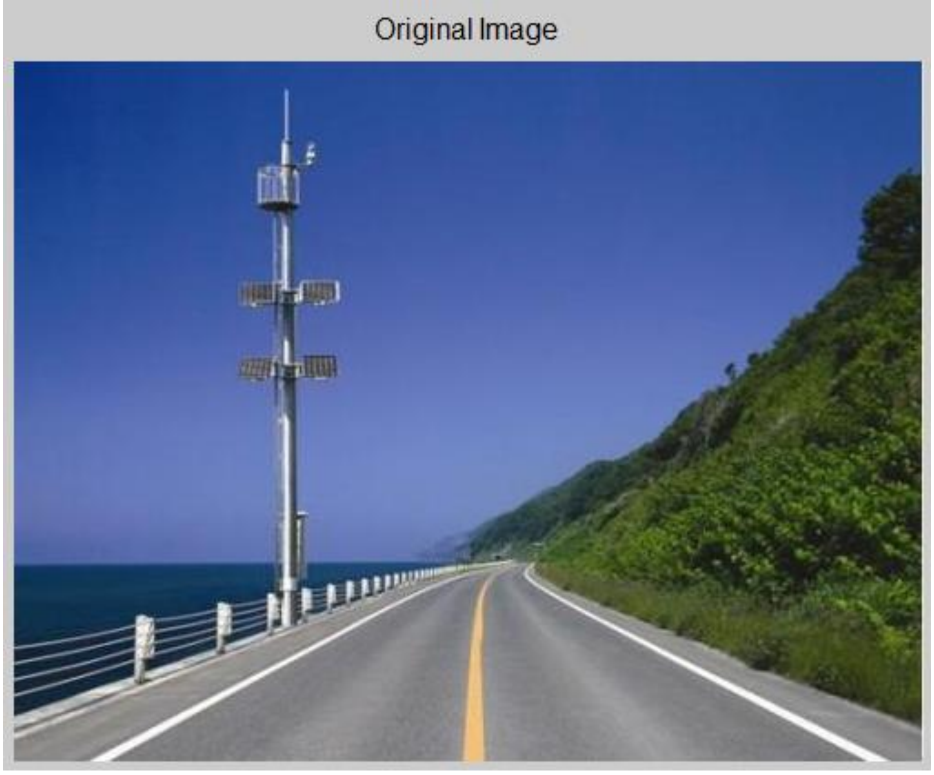

Figure 5.22. Empty road (3)

Figure 5.22 is the third empty road image. In this image there are mainly green trees and blue sea. There is only one high-rise transmitter tower. The horizontal axis's brightness values (Figures 5.23 and 5.24) reflect this high-rise transmitter tower. It is clear that the grate filter brings lower brightness values than the bar filter, which makes the computers decision more accurate. At last, the decisions from both filters are "Object not detected". Hence the original image can be used to update the background.

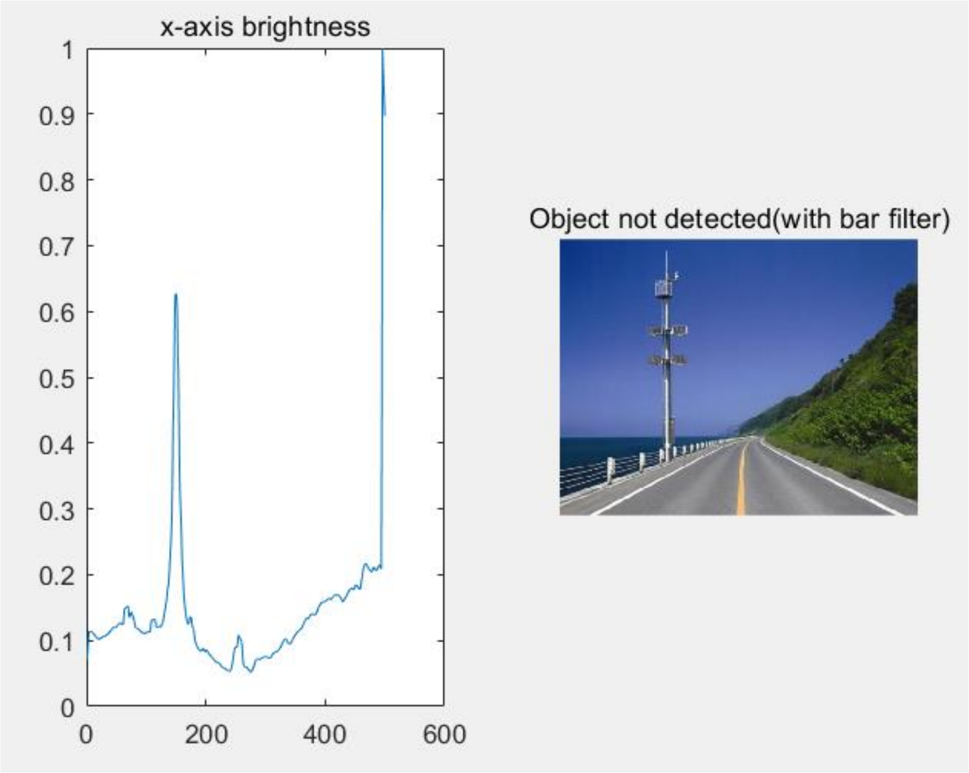

Figure 5.23. The horizontal axis's brightness values and searched result (with bar filter)

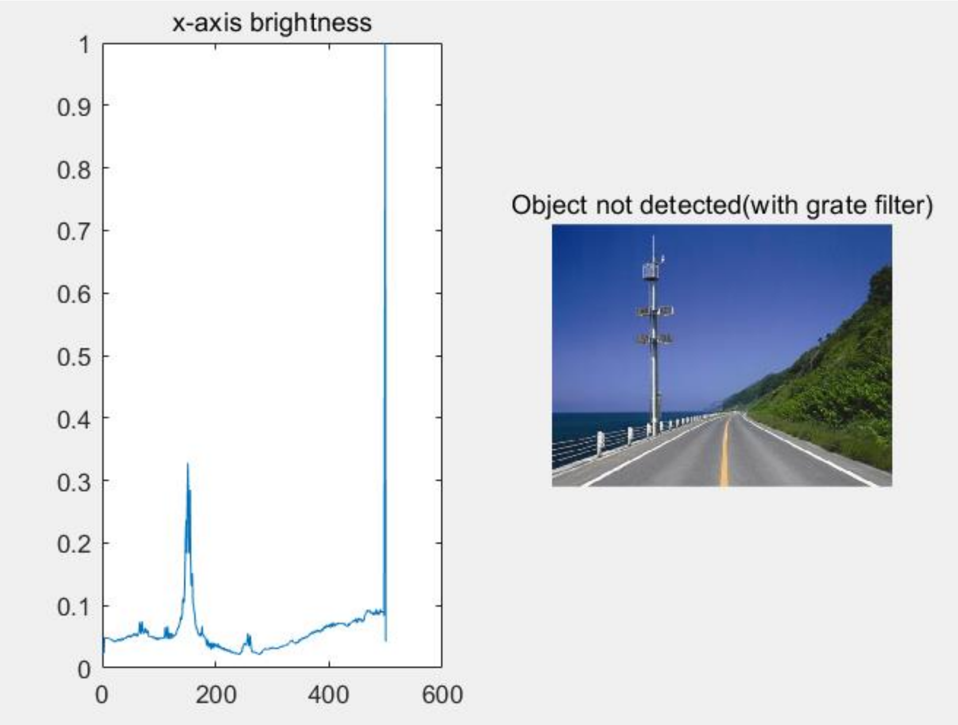

Figure 5.24. The horizontal axis’s brightness values and searched result (with grate filter)

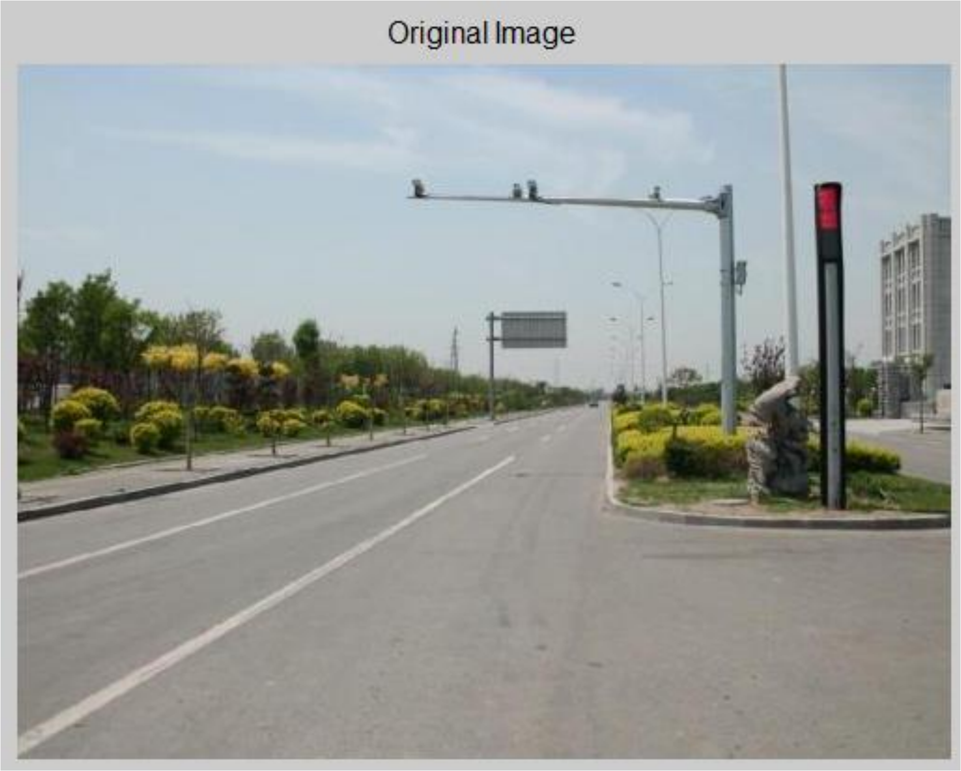

Figure 5.25. Empty road (4)

Figure 5.25 shows the fourth empty road. There are several posts crowded installed. From Figures 5.26 and 5.27, we can find several narrow high frequency areas, and all these areas can be recognized individually. Again the grate filter shows more details on the brightness values about the posts. For every one area, it cannot meet our threshold demand. In this case, they are all judged as noise and the decisions from both the filters are correct, which are "Object not detected". Since there is no object identified inside this image, it can be used to update the background.

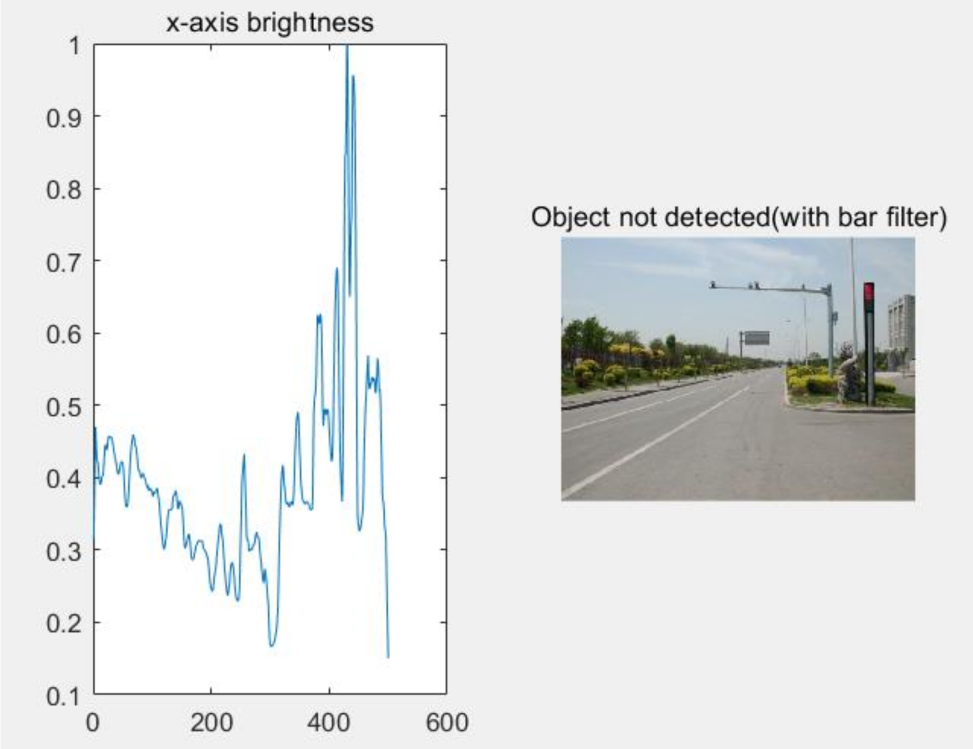

Figure 5.26. The horizontal axis's brightness values and searched result (with bar filter)

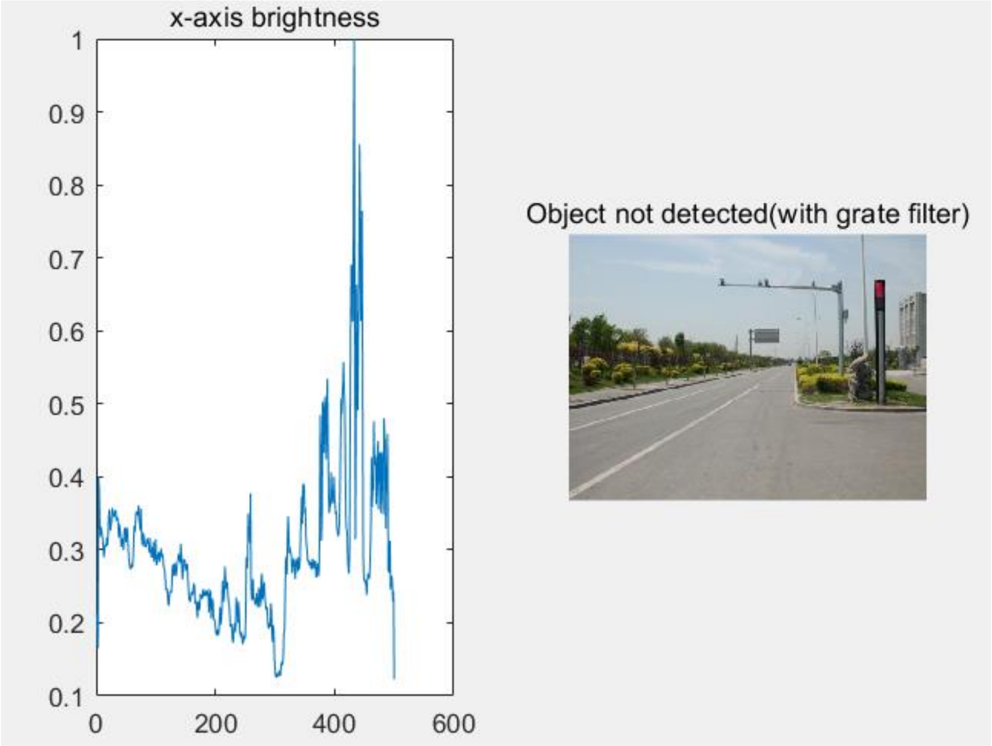

Figure 5.27. The horizontal axis's brightness values and searched result (with grate filter)

5.5 Summary

This chapter proposed a new grate filter, which is used to identify whether there are objects in these images captured by the cameras in urban surveillance systems. In this way the background images can be updated from time to time when no object is detected. It is a low-complexity solution helping to set up the background image for urban surveillance systems, hence to make the foreground detection process easier. This method does not need complex computing for background detection. Furthermore, it is robust to the impact of light changing and does not depend on the moving situation of objects. We tested and compared the proposed grate filter with the bar shape filter. The results show that the grate filter can obtain more details on the intercorrelation results of the images with lower computational load.

CHAPTER 6

FAST SALIENCY-AWARE IMAGE HASH FOR NEAR DUPLICATED IMAGES RETRIEVAL

Object recognition is an important subject of intelligent systems. These systems facilitate the problem of automated identification. Image Hash is an important tool to detect objects in images, but it encounters great difficulties when the objects are slightly moved. In this chapter, we combine image Hash method with our proposed density scan method. In this way, the slight moving of the object (i.e. saliency) will be detected and the near duplicated images with same source will be determined automatically. By using this simple algorithm, our proposed image Hash method is fast, saliency-aware and practical.

6.1 Introduction

Hash methods have been widely used in image search because of their two benefits: less memory usage and high search efficiency [81]. In image Hash methods, unique codes which are extracted from images are used as the identifications for images. This method is widely used in the field of image searching/retrieval since it can efficiently identify the duplicate copies of images [82].

With the wide use of image editing, image Hash based tampering detection methods have been widely studied. There are three most commonly used image Hash algorithms, they are average hash (AHash), perceptive hash (pHash) and difference hash (dHash). A lot of research work has been done on these hash methods. For example, a method based on discrete wavelet transform was proposed and it is robust to different content-preserving and other common manipulation [83]. In paper [84], the authors proposed a quaternion based image hashing to detect many types of tamping. Authors in paper [85] proposed a multi-scale image hash method for tampering detection. In paper [86], the authors improved image hash method using both global feature (based on Zernike moments representing the luminance and chrominance characteristics of an image) and local feature (include position and texture information of salient regions in the image). Authors in [87] proposed a perceptual image hash method to detect image tampering and tampering localization.

Although there are already a lot of research successes, the basic principle of image Hash methods determine that the structure of the images cannot be tampered, i.e., the positions of the main objects inside an image should be constant. This character seriously limits the wide usage of image Hash. Hence the saliency detection is an important process to find the important contents inside images. Saliency detection aims at identifying the main objects in images.

There are many researches in the image salient area. In paper [88], the authors proposed a new context-aware saliency because in some cases the context of the dominant objects is as the same importance as the objects. Paper [89] constructed a large image database so that to learning to detect salient objects. Since hundreds of computational saliency models have been proposed, the authors compared 29 salient object detection models in [90].

In [80], we found that an object is normally the highest density part of an image. Hence, we proposed a novel low-complexity automatic object detection method. Here we name our proposed method as Density Scan method. In this chapter we will combine image Hash method with our Density Scan method to find near duplicate images. As far as we know, we are the first to solve object position tampering problem for image Hash techniques.

6.2 Solving Object Position Tampering Problem

6.2.1 Tampering Images and Pre-treatment

We know that there is huge number of images spread on internet every day. When an image travels on different websites, it is often modified, sometimes part of it (usually the most important part) be cut out and then continues its journey. Image Hash is an important and efficient method to search similar images on internet, but it can do nothing to help this kind if image modification. In this chapter, we use the average Hash (aHash) to combine with our Density Scan method to solve this kind of problem. When using aHash, we uniformly resize all the input images to 10*10 and convert the resized images into grayscale ones. In this case the hamming distances for the images are calculated as the difference between two 100 long sequence of numbers. The hamming distances are automated shown on the titles of the images.

Figure 6.1 shows two images, which is very easy to be found that they are come from the same source. From the view of our human eyes, there is only a little modification. It is very clear that since there is only little difference between the two images, the most important information (car plate) is not destroyed. Unfortunately, because of this little change, when using image Hash method, the hamming distance is calculated as 6, which is a big enough number to show that the 2 images are different images from the view of computers (considering there are 6 difference numbers inside 100 numbers).

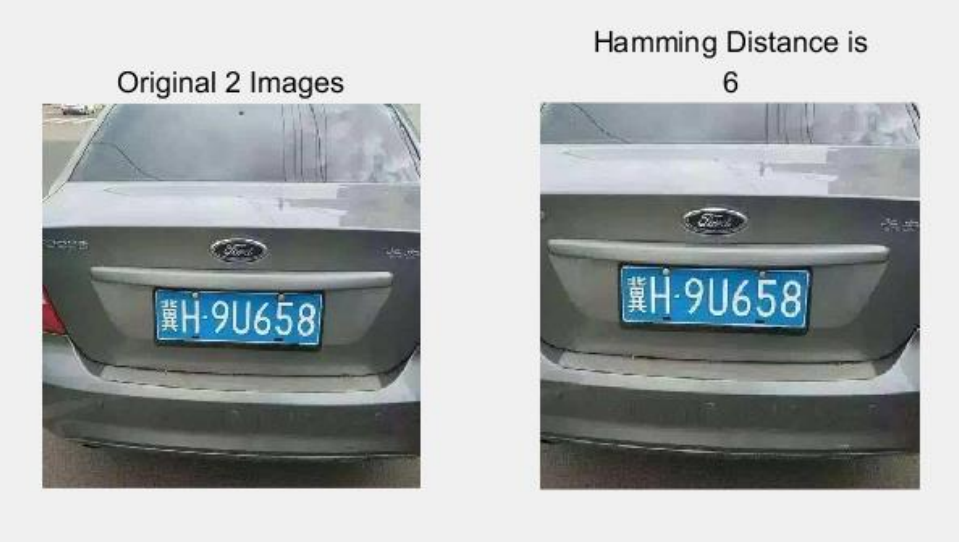

Figure 6.1. The original car images and there hamming distance

Then we will begin our pre-treatment. First we transfer the original 2 images into grey ones, and then calculate the gradients of the grey images. Considering every two-dimension grey image as a two-dimension matrix P, we can easily get the numerical gradients of these matrixes by defining P_x and P_y as the differences in horizontal direction and vertical direction. The gradient for every pixel is shown below:

$$P = \sqrt{{P_x}^2 + {P_y}^2} \qquad (6.1)$$

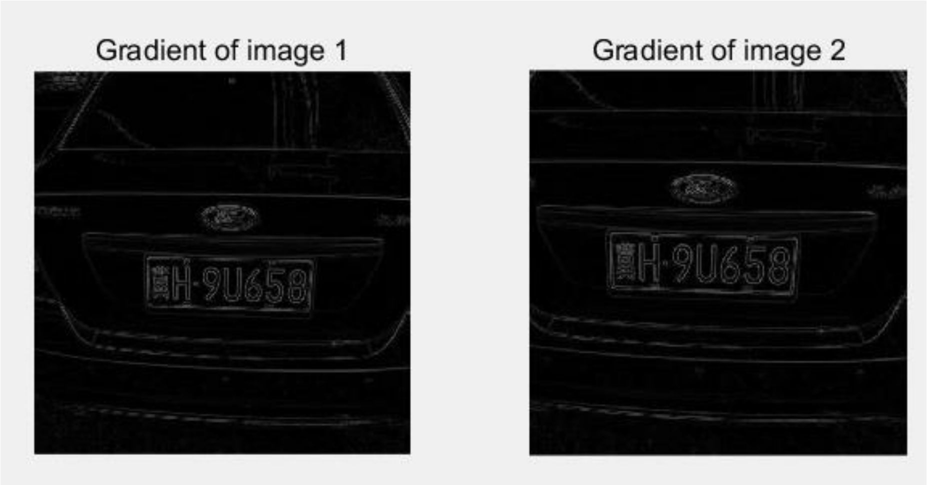

Figure 6.2. The Gradient of the Grey Images

The gradient result is shown in figure 6.2. From these figures, we can find that the plate areas are conspicuous in our human's eyes. It is because the plate areas are the most density parts of the images and catch our attention, which simultaneous hold the most important information since plates reveal car information. Our aim is to pick out the car plate areas with our novelty method and compare them using aHash method. In this way the computer will find that they are coming from the same source.

6.2.2 Our Density Scan method

After investigating the character of the car plates, we found that these areas hold the characters of high density comparing with the whole images. In paper [80], we proposed a novel method to pick out the high density area of an image.

As we already know, the function for intercorrelation for the two-dimensional discrete data is expressed as below:

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(i+m, j+n)K(m,n) \qquad (6.2)$$

This formula reveals that the intercorrelation for two-dimensional images is the addition of their individual multiplications. Since the high density area of an image reveal more high value ('1' present write and '0' present black in a grey scale image), we designed a matrix filter (all numbers are set as 1s to constitute this filter) to filter the object image. In this way, the high density area will be shown off and can be chose to be presented by only a one dimensional series. Fundamentally, our proposed method is a descending dimension algorithm.

The perpendicular matrix is used to filter the image in horizontal way and the horizontal matrix is used to filter the image in perpendicular way. The diagrammatic sketch of this filter is shown in figure 6.3. We choose the matrix filters as wide as 20 columns to smooth noise.

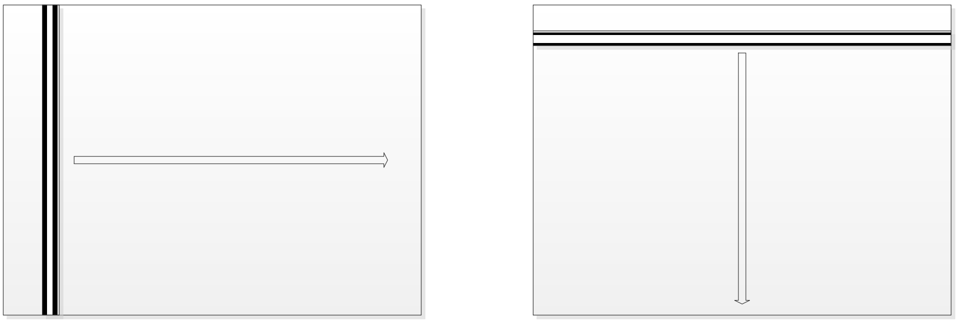

Figure 6.3. Matrix filters to scan object images

After filtering the 2 object images (gradient images) in horizontal way, we get the brightness images. The high density areas show brighter results and have centralization trends in the y-axis'. Then we plot the one-dimension figures by choosing the x-axis brightness value when the y-axis value is chose at middle of the brightness images. In this way, the two-dimension brightness images are reduced to one-dimension series. The filtered brightness images and the one-dimension brightness series are shown in figure 6.4 and 6.5.

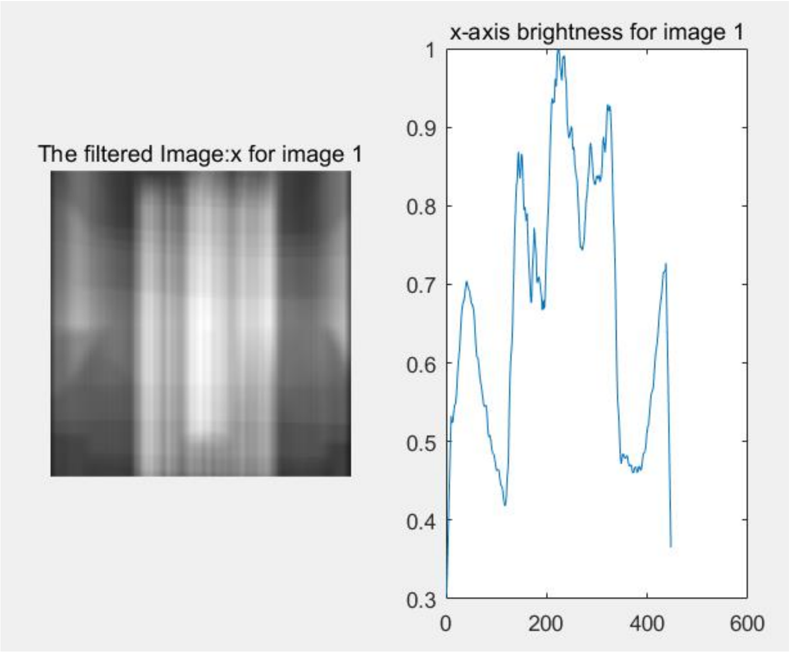

Figure 6.4. The filtered brightness results of image 1 (x-axis)

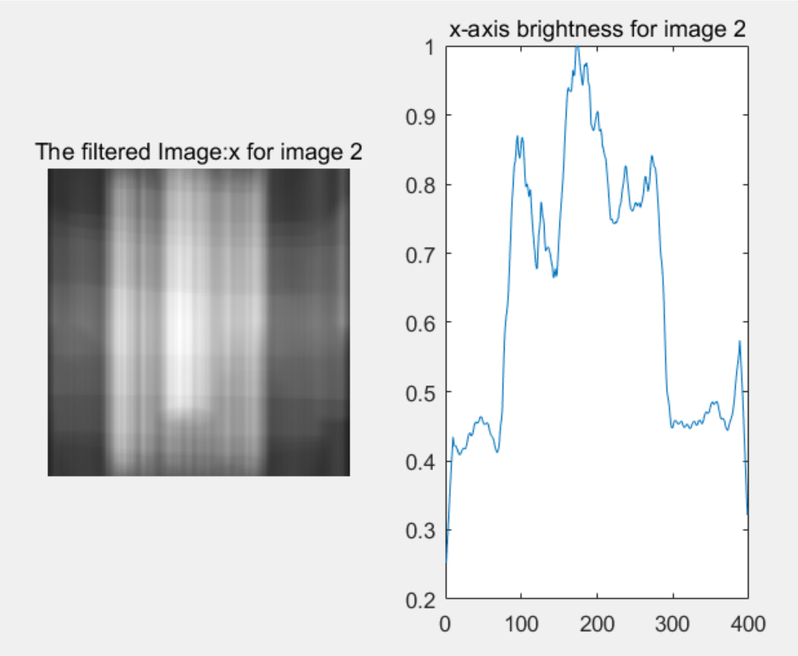

Figure 6.5. The filtered brightness results of image 2 (x-axis)

Since the car images have already be converted into one-dimension brightness values, we only need to search these brightness values from left to right and simply choose the threshold at 0.5. In this way, the x-axis' edges are got. We cut the corresponding horizontal areas from the original images, and the two results are shown in figure 6.6. Considering we only used a kind of simple filter to get these results, they are satisfying.

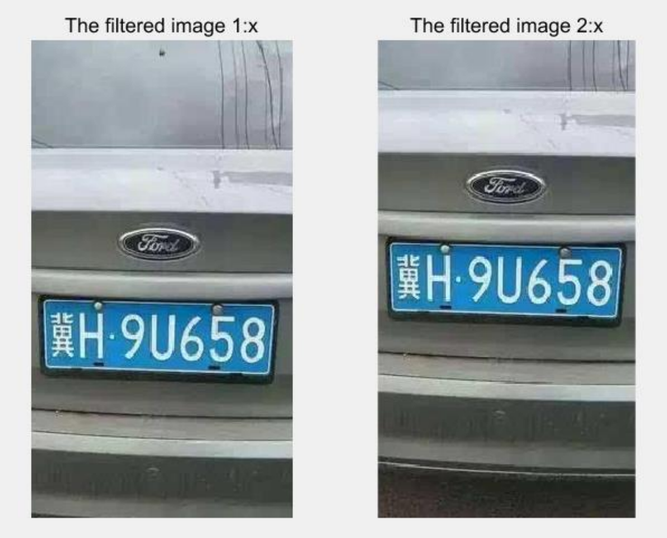

Figure 6.6. The filtered horizontal areas of the 2 images

Similar to the horizontal direction, we use our matrix array filter in the perpendicular position. We only need to filter the pre-treated image areas in figure 6.6 to reduce calculation. Again we choose the width of the filters as 20 rows to filter the object images. The results are shown in figures 6.7 and 6.8. It is obviously that the high density areas are shown by brighter areas. This time they have centralization trends in the x-axis'.

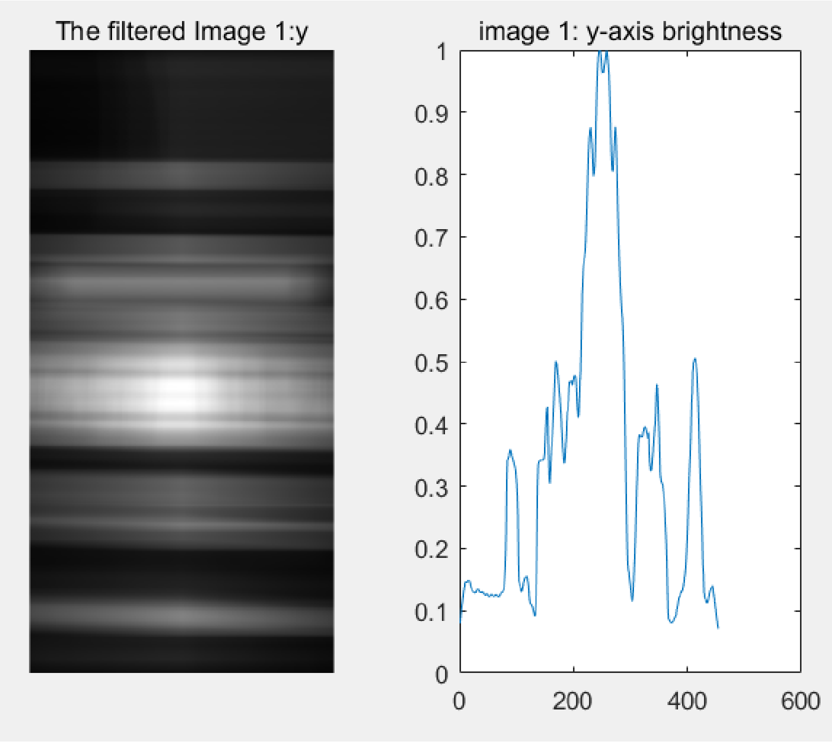

Figure 6.7. The filtered brightness result of image 1 (y-axis)

After plotting the one-dimension figures by choosing the y-axis brightness values while the x-axis value is chose at middle of the filtered brightness images, the one-dimension figures for straightforward view are got as well. The results are also plotted in figures 6.7 and 6.8.

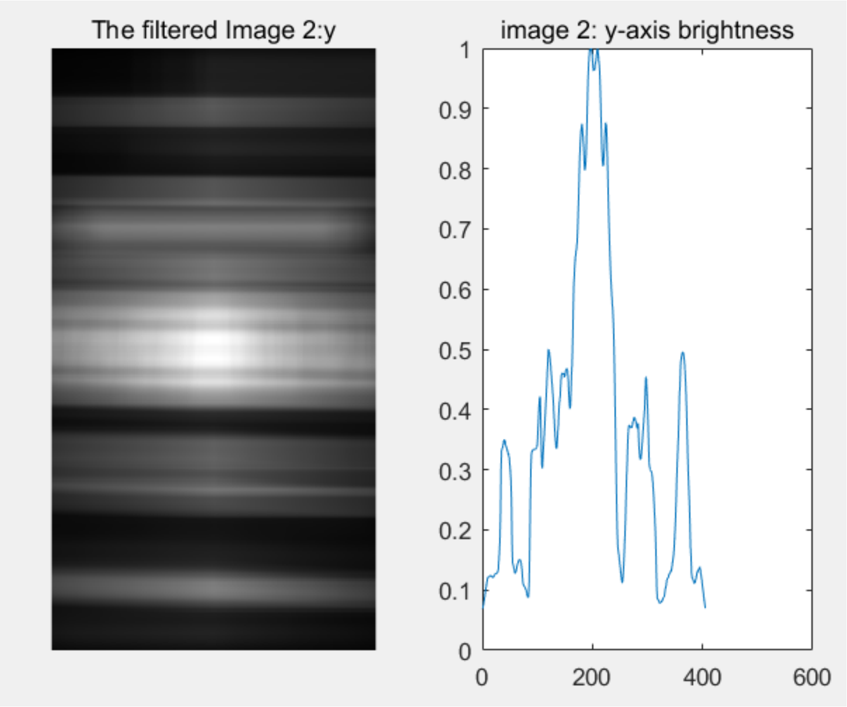

Figure 6.8. The filtered brightness result of image 2 (y-axis)

Again we search these one-dimension values from left to right and choose the threshold brightness value as 0.5 to get the y-axis edges. After cutting the corresponding perpendicular areas from the original images, the final results are shown in figure 6.9.

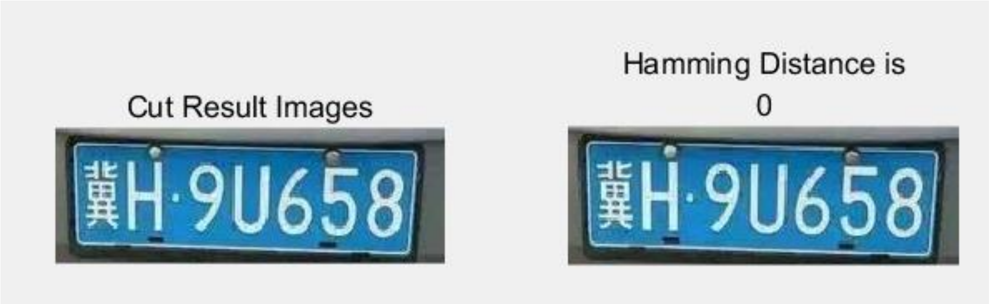

Figure 6.9. The final cut 2 images and their Hamming Distance

The hamming distances for the images are calculated and automated shown on the titles. The hamming distance is 0 in this case, which means the 2 result images are exactly the same images, judged by aHash method.

We know that the 2 original images are come from the same source. There is only a little cut modification which leads to the erroneous judgement when image Hash method is utilized. After finding out the most density parts of the images, image Hash methods can become our valuable tools again in the image searching areas.

6.2.3 Further Testing on Our Proposed Method on Car Images

In this section, we test our method to see how it works on more car images. Figure 6.10 are example original car images. Although they look very similar, the hamming distance is calculated as 17. Obviously the 2 original images are come from the same source and there is only a little moving modification. But hamming distance 17 means they are 'not' similar.

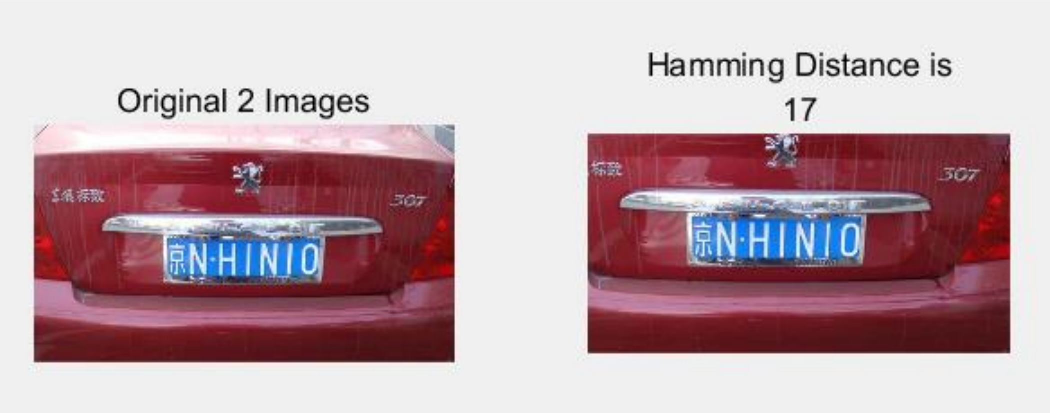

Figure 6.10. The example original car images and there hamming distance (1)

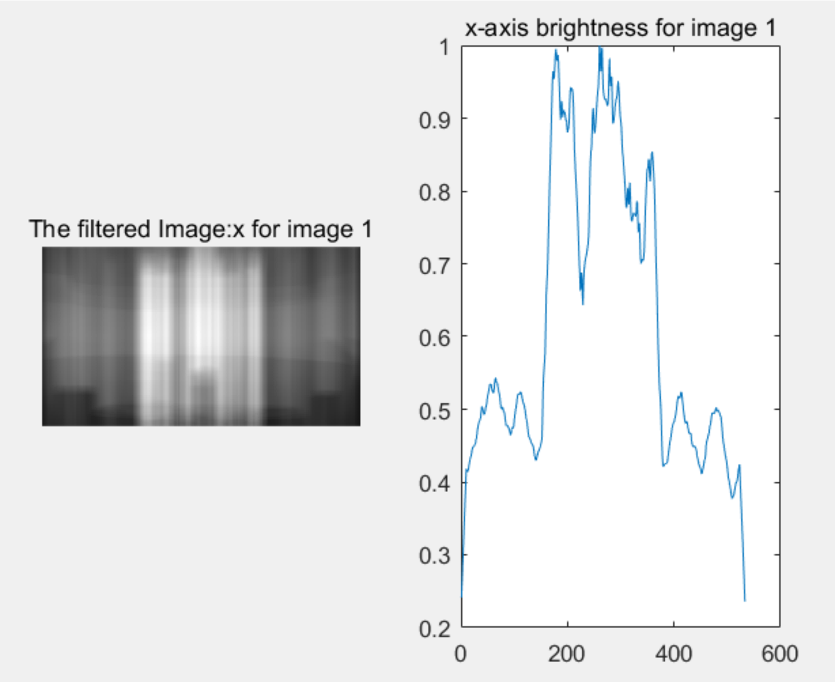

Figure 6.11. The filtered brightness result of image 1 (x-axis)

Figure 6.11 and 6.12 are the filtered brightness value results of image 1 and 2 in horizontal way and we plot the one-dimension brightness figures. On the reduced one-dimension series, we search from left to right with threshold 0.5. Then the x-axis edges are got corresponding to the original images.

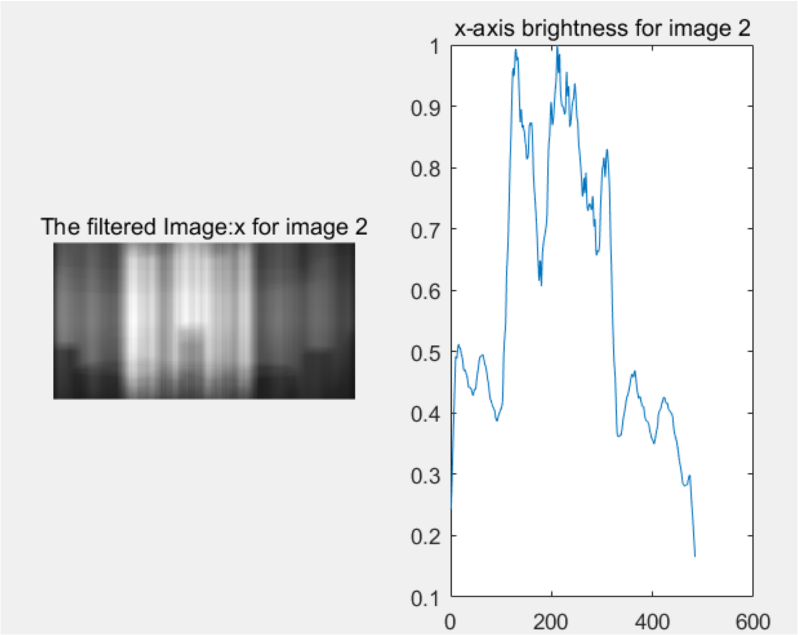

Figure 6.12. The filtered brightness result of image 2 (x-axis)

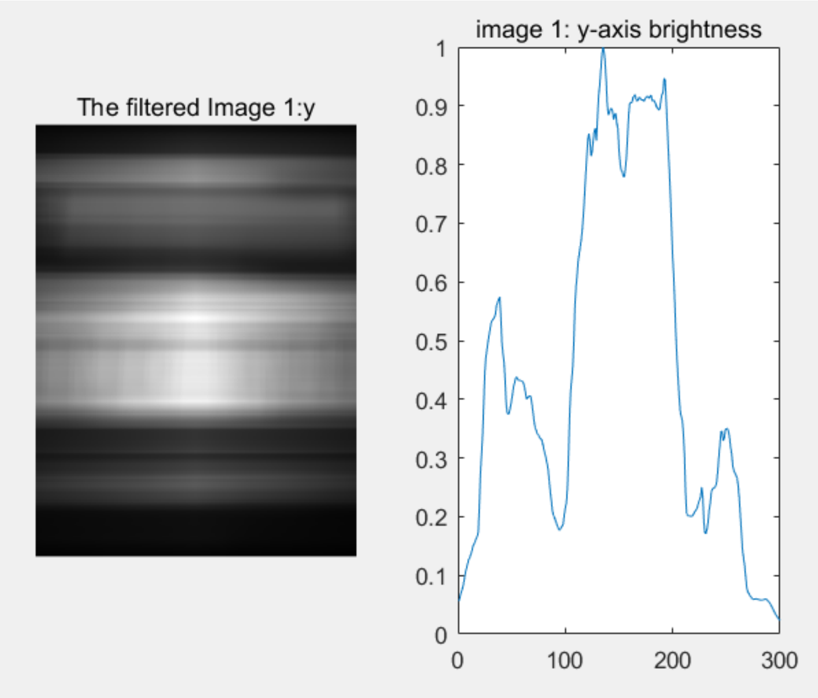

Figure 6.13. The filtered brightness result of image 1 (y-axis)

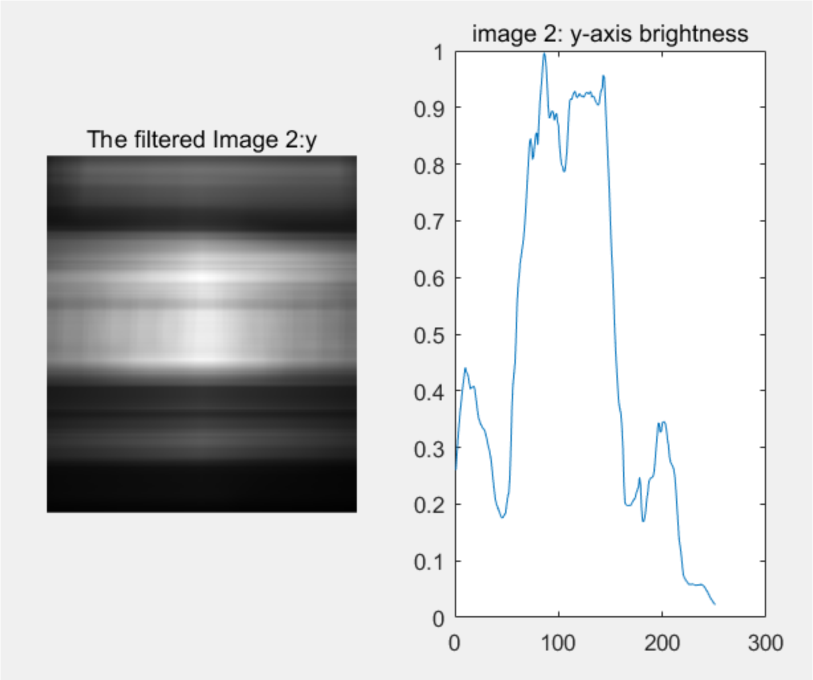

Figure 6.14. The filtered brightness result of image 2 (y-axis)

Figure 6.13 and 6.14 are the filtered brightness value results of image 1 and 2 in perpendicular way (within the chose x-axis area). Again we plot the one-dimension brightness values and get the y-axis edges by choosing threshold 0.5. The final cut results are shown in figure 6.15.

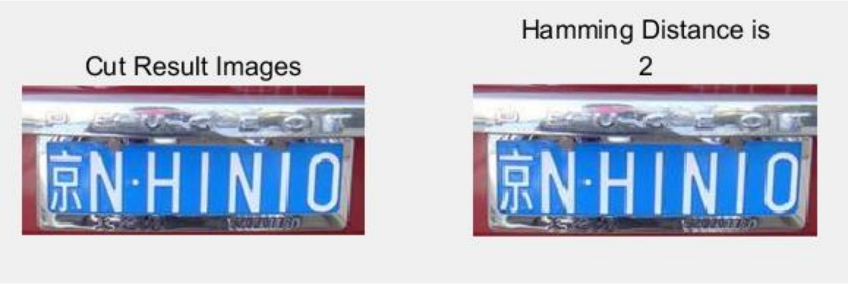

Figure 6.15. The final cut 2 images and their Hamming Distance

By using aHash method, the hamming distances for the final cut images are calculated as 2, since the hamming distance is so small, the 2 result images are judged as same. It is clearly that filtering out the most density parts of the images are very helpful for image Hash methods to find the same source images when cut and move modifications were used on them.

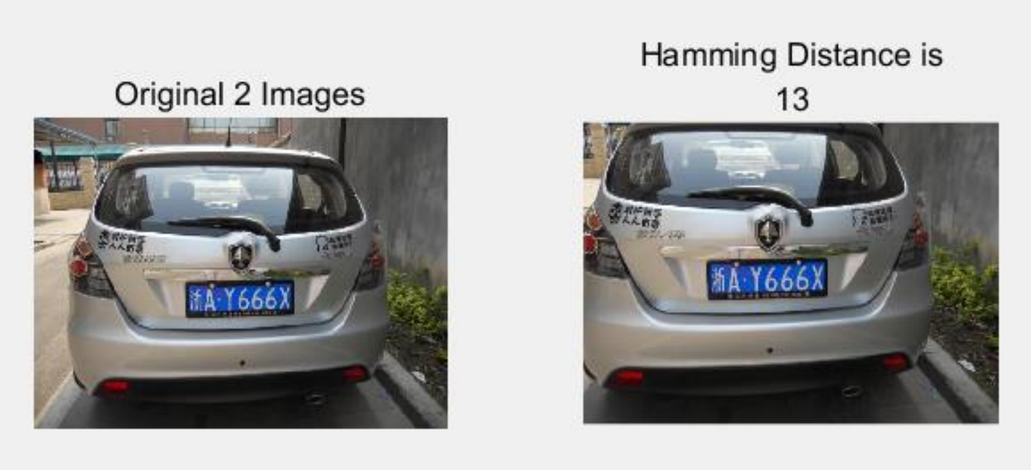

Figure 6.16. The example original car images and there hamming distance (2)

Figure 6.16 is another example. The hamming distance between the 2 images is 13. Figures 6.17 to 6.20 are the middle process of filtering brightness values and the one-dimension brightness values. The final cut results are shown in figure 6.21.

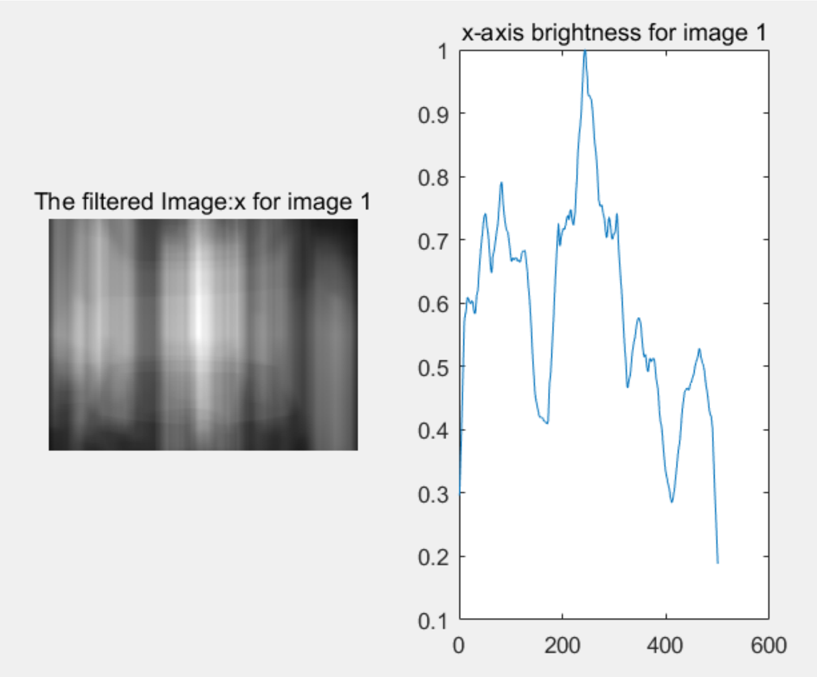

Figure 6.17. The filtered brightness result of image 1 (x-axis)

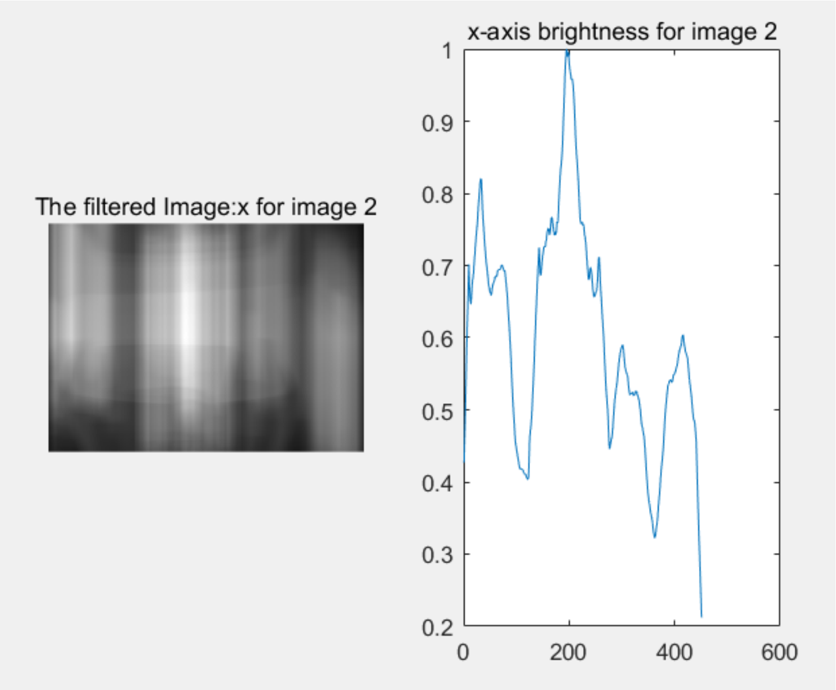

Figure 6.18. The filtered brightness result of image 2 (x-axis)

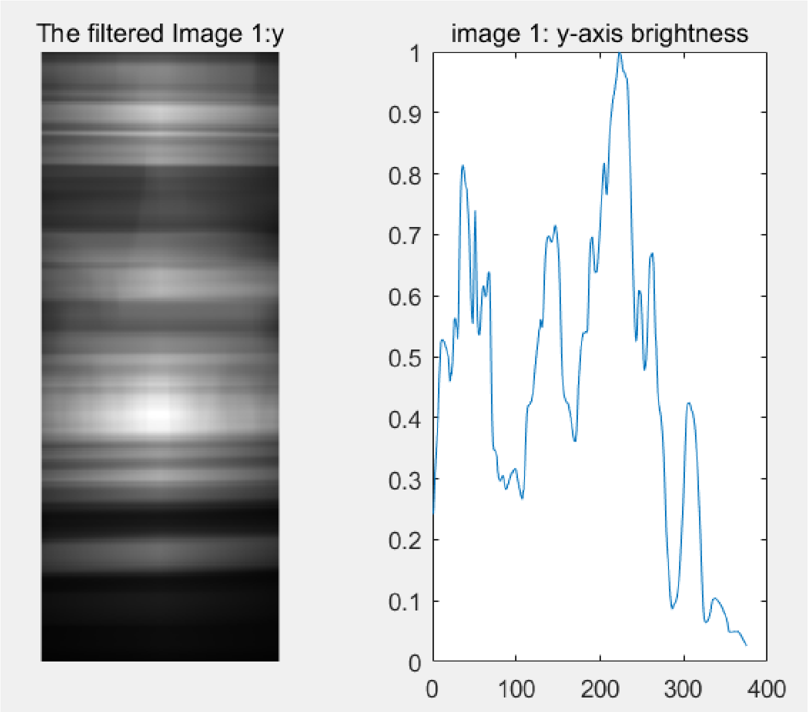

Figure 6.19. The filtered brightness result of image 1 (y-axis)

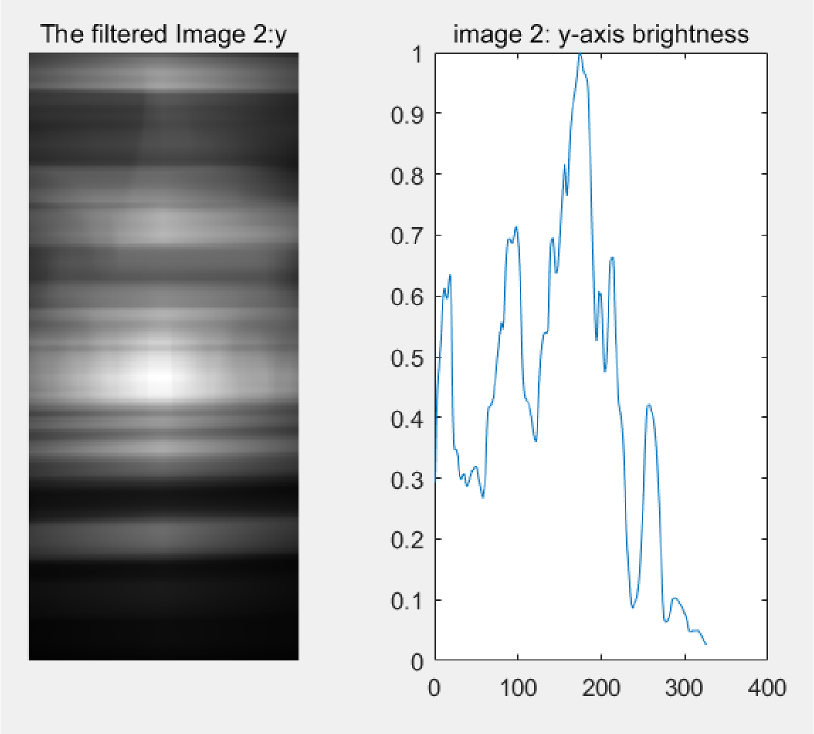

Figure 6.20. The filtered brightness result of image 2 (y-axis)

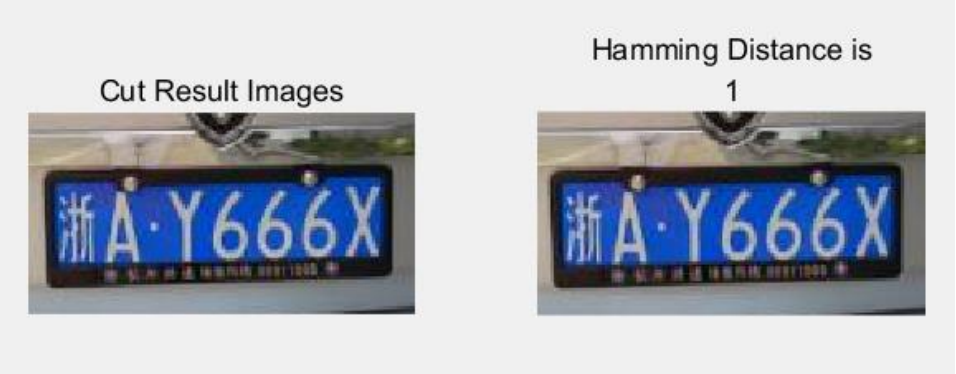

Figure 6.21. The final 2 images and their Hamming Distance

The hamming distance for the final images is only 1. This example is another proof to indicate that filtering out the most density parts (car plates) of car images is a powerful method to help image Hash searching processes.

6.3 The Flow Chart of Our Method

The flow chart of our proposed method is shown in figure 6.22. At the beginning of the comparison, image aHash method is used to calculate the hamming distance of the 2 original images (with slight difference). After using our method to pick out the most density area of an image, we get 2 new images. Image Hash method is used on the 2 new images and their hamming distance is calculated. We found that the new hamming distance is much smaller than the prior one. The results indicate that our algorithm is a simple but effective way to find the similar images coming from the same source.

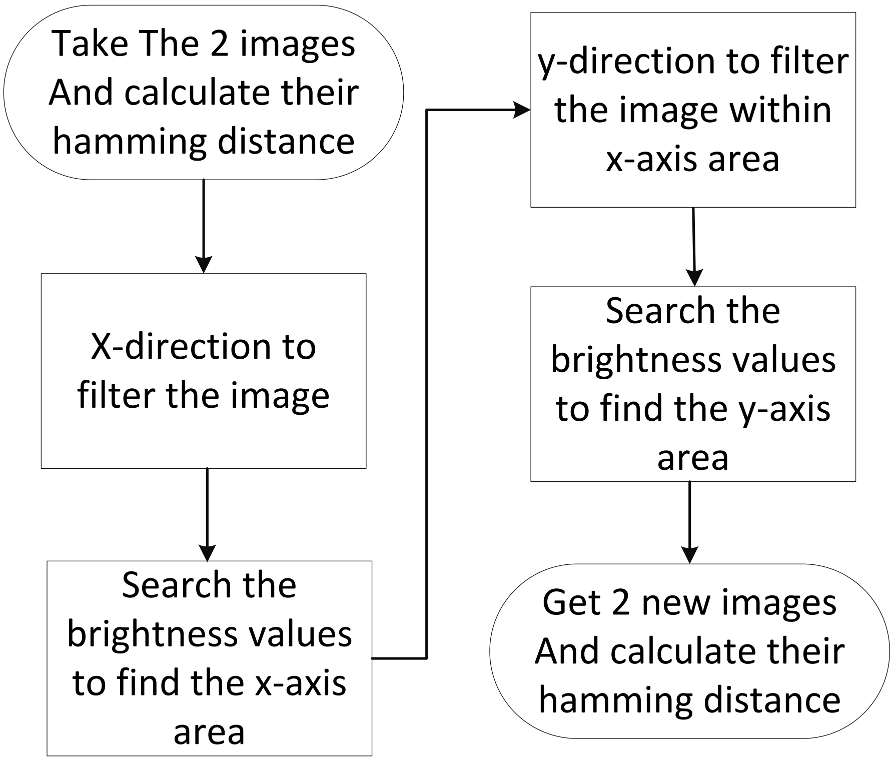
Take The 2 images And calculate their hamming distance
X-direction to filter the image
Search the brightness values to find the x-axis area
y-direction to filter the image within x-axis area
Search the brightness values to find the y-axis area
Get 2 new images And calculate their hamming distance

Figure 6.22. The process of our proposed method

6.4 Experimental Results

In this section, we carry experiments to test the usage of our algorithm to more kinds of images. Figure 6.23 is an example of one cat. Although the 2 images show the same cat, the little position modification make the hamming distance between them 6, which reveals their difference.

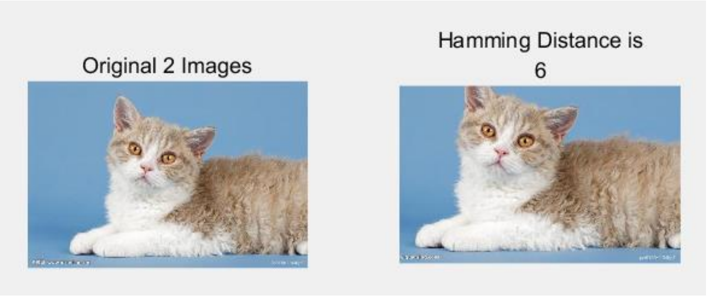

Figure 6.23. The example original car images and there hamming distance (1)

Figures 6.24 to 6.27 are the filtered brightness value results of the 2 images in horizontal and perpendicular directions and then we plot their one-dimension figures.

The threshold brightness values are also chosen at 0.5. Then the x-axis and y-axis edge values are got corresponding to the original images. The final cut results are shown in figure 6.28.

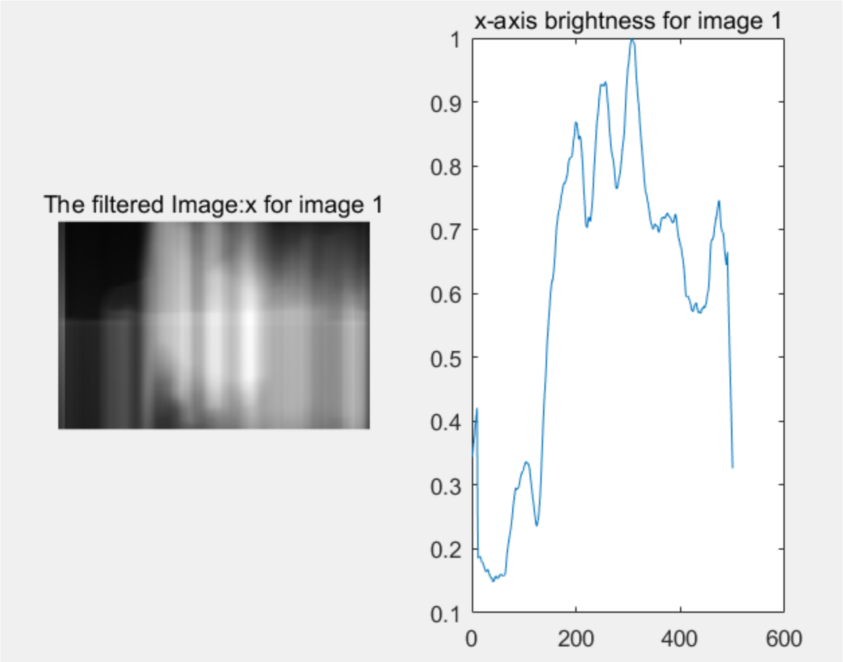

Figure 6.24. The filtered brightness result of image 1 (x-axis)

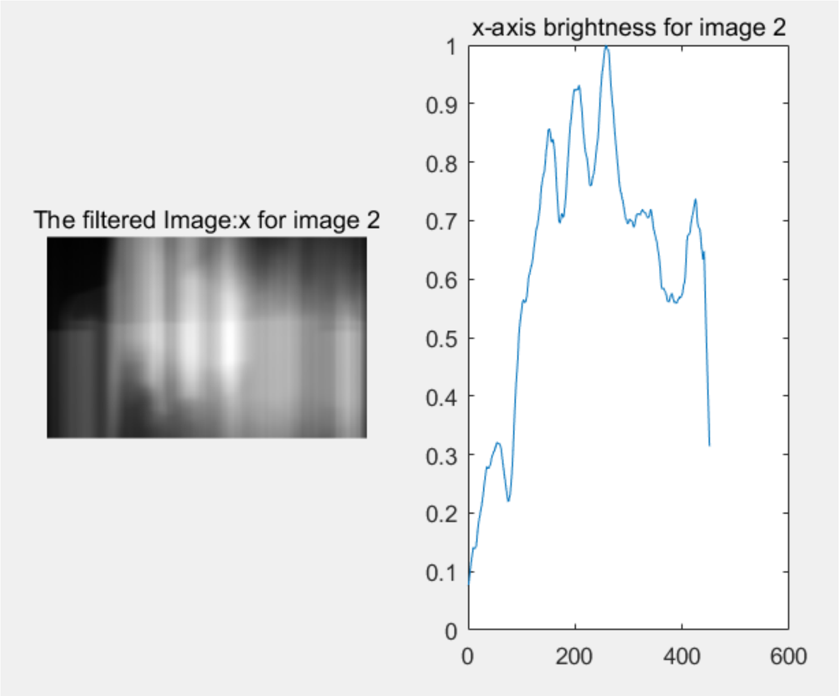

Figure 6.25. The filtered brightness result of image 2 (x-axis)

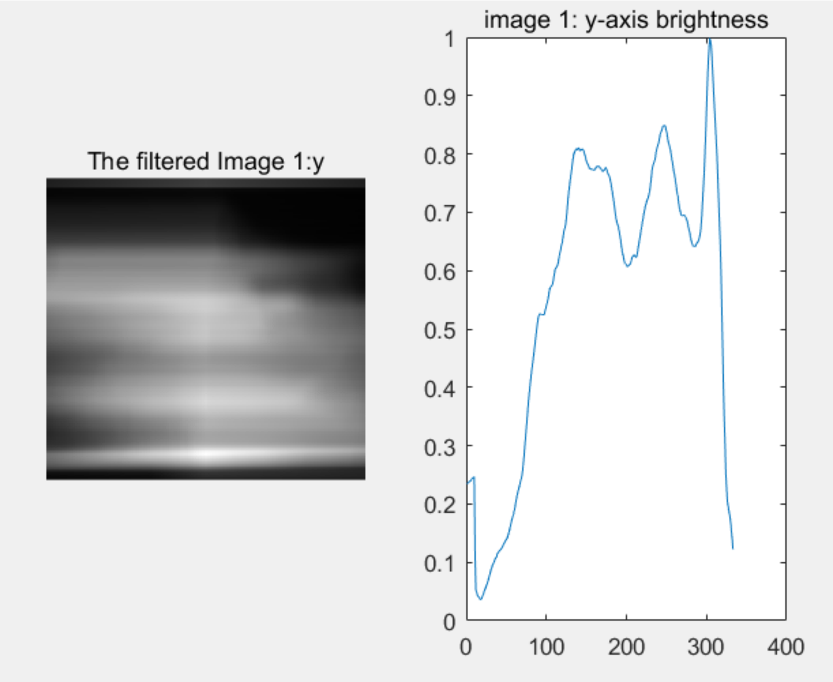

Figure 6.26. The filtered brightness result of image 1 (y-axis)

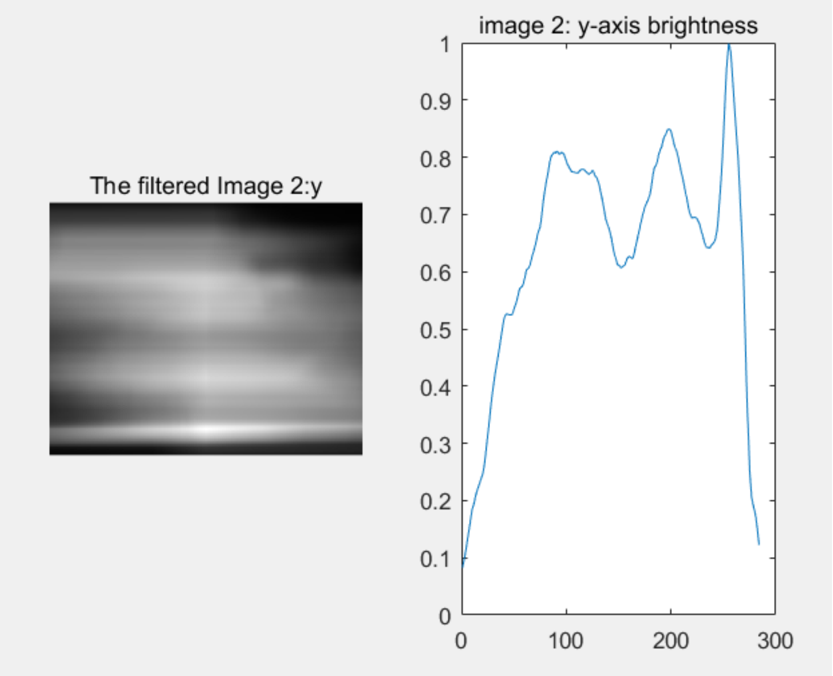

Figure 6.27. The filtered brightness result of image 2 (y-axis)

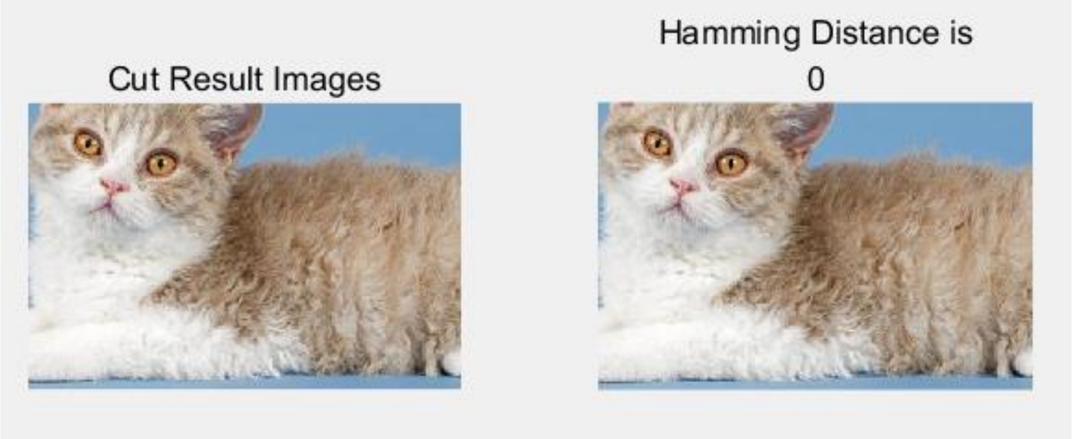

Figure 6.28. The final results of 2 cat images and their Hamming Distance (1)

The hamming distances for the final 2 cat images is 0, which means that the 2 result images are exactly same images, judged by the computer with image aHash method.

Figure 6.29 is another cat example. Although they looks exactly the same cat, position difference make the hamming distance between the 2 images 13, which is fairly ‘far’ for hamming distance.

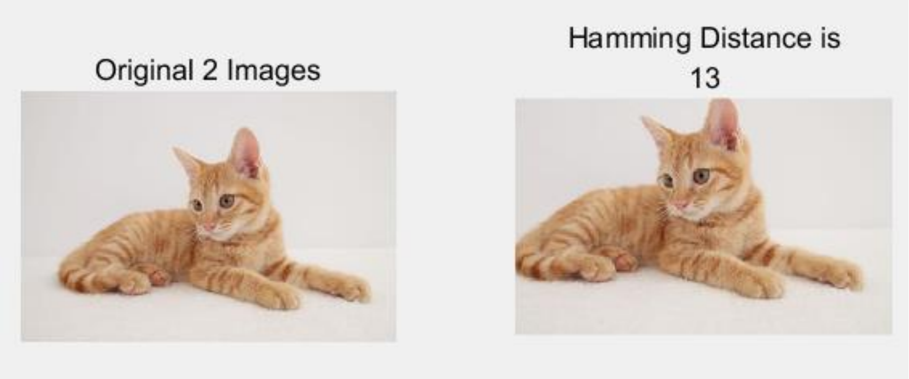

Figure 6.29. The example original car images and there hamming distance (2)

Figures 6.30 to 6.33 are the filtered brightness value results of the 2 images in horizontal and perpendicular directions and their one-dimension figures. The threshold brightness values are again chosen at 0.5. After the x-axis and y-axis edge values are got corresponding to the original images, the final results are shown in figure 6.34.

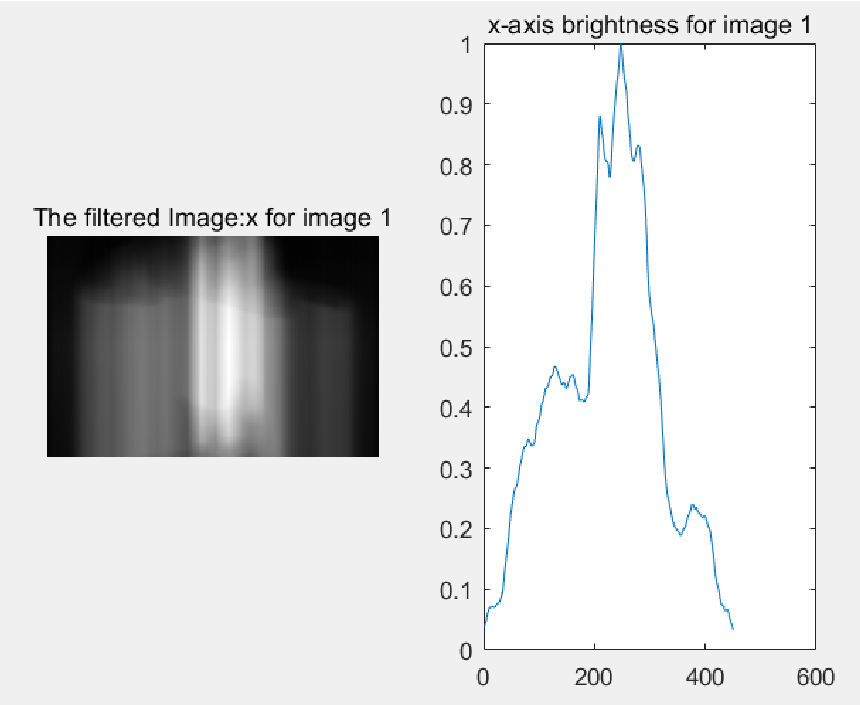

Figure 6.30. The filtered brightness result of image 1 (x-axis)

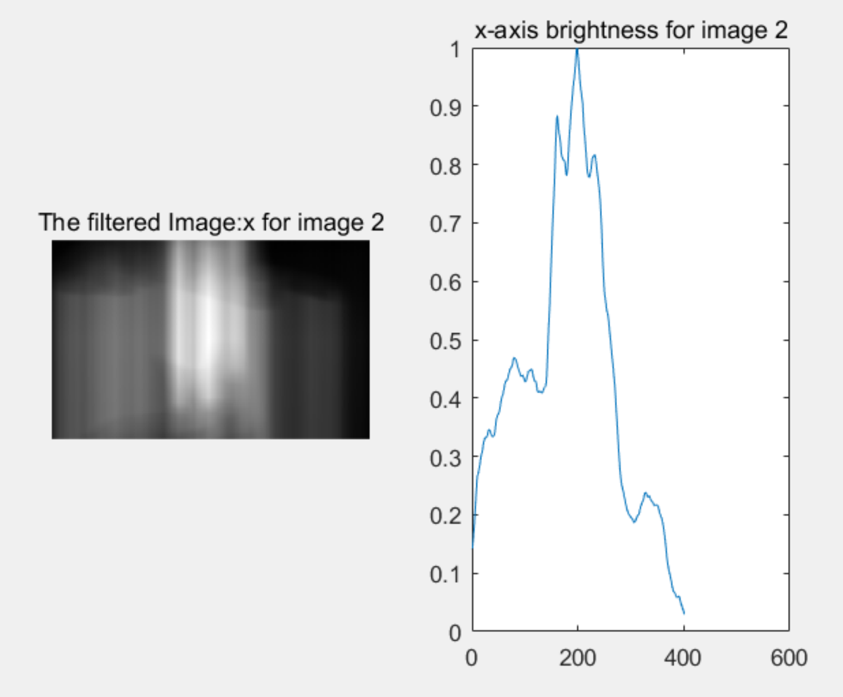

Figure 6.31. The filtered brightness result of image 2 (x-axis)

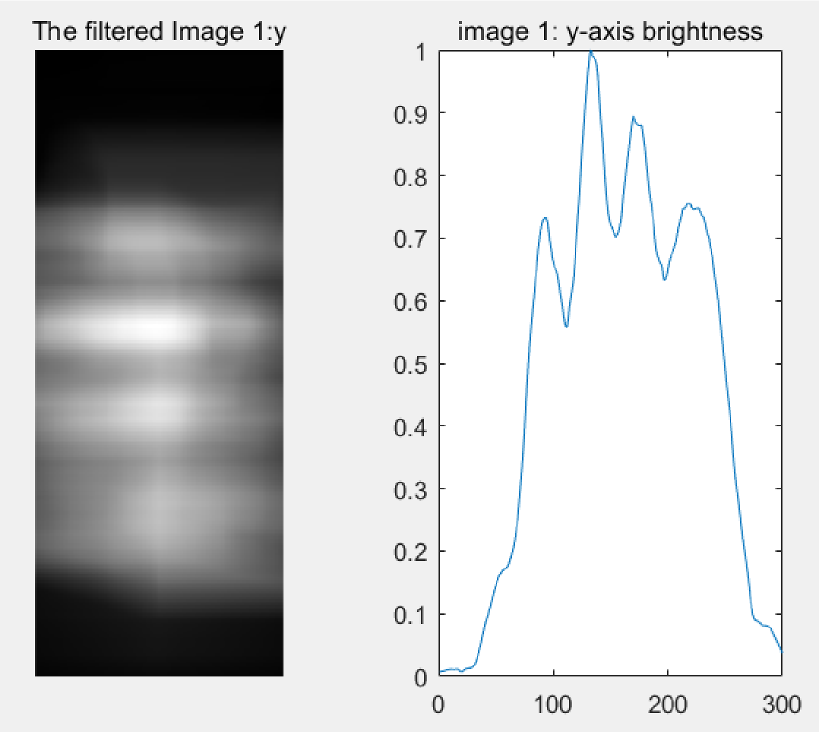

Figure 6.32. The filtered brightness result of image 1 (y-axis)

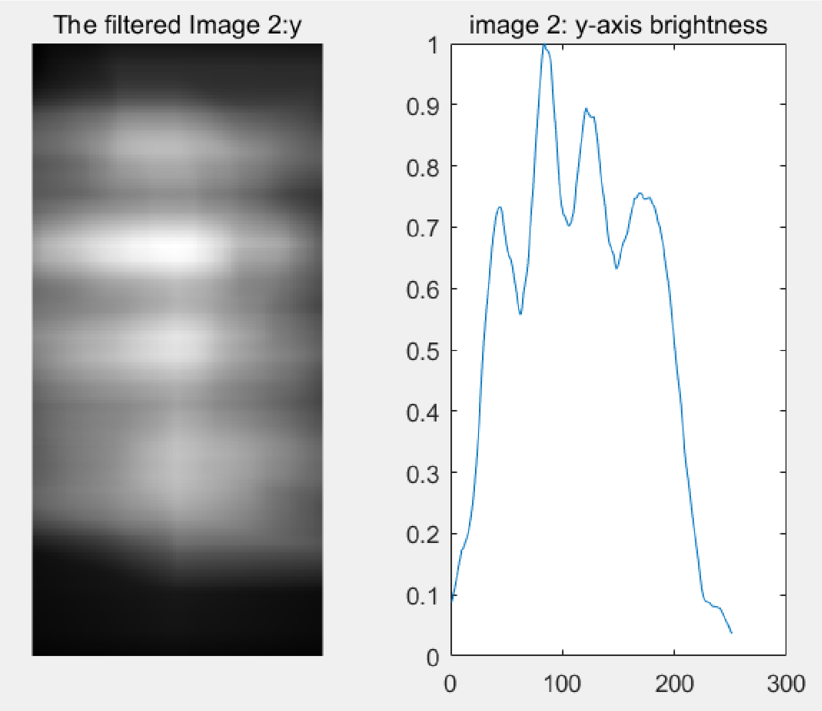

Figure 6.33. The filtered brightness result of image 2 (y-axis)

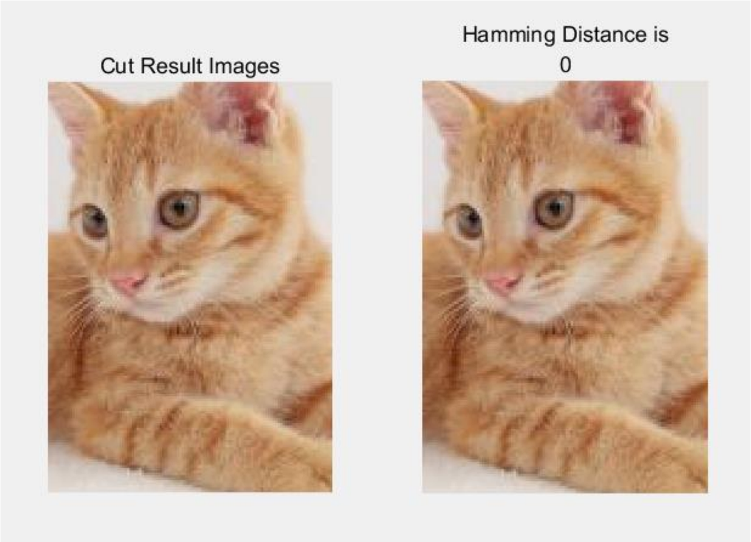

Figure 6.34. The final results of 2 cat images and their Hamming Distance (2)

The hamming distances for the final cut results of the 2 cat images is 0, which means that the 2 result images are again exactly the same images.

In the above examples, we expand our objects from cars to cats and we find that filtering out the most density parts of the images is still useful to combine with image Hash methods; in this way the image Hash method is easy to find that they are in fact the same cat.

Then we further expand our examples to birds. In figure 6.35, the hamming distance between the 2 bird images is 7; hence they are 'different' images.

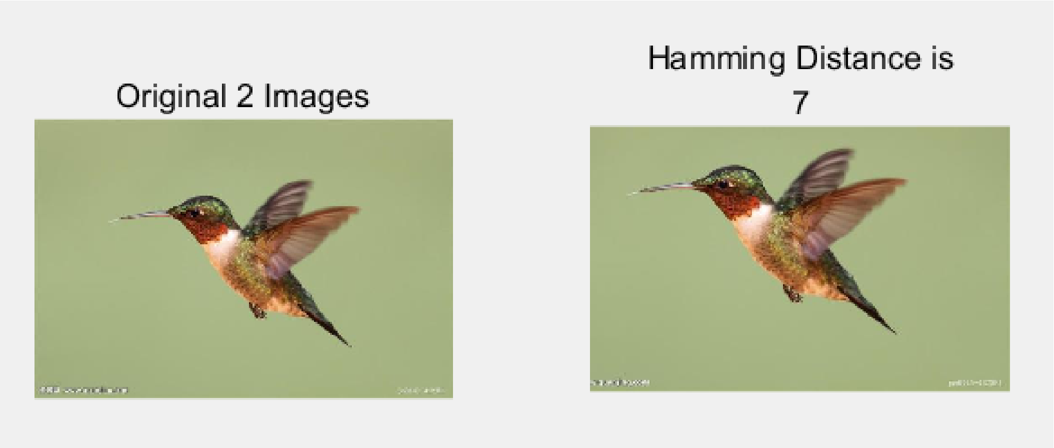

Figure 6.35. The example original bird images and there hamming distance (1)

Figures 6.36 to 6.39 are the filtered brightness value results of the 2 birds' images in horizontal and perpendicular directions and their one-dimension figures respectively. On the one-dimension series, we also search with the threshold brightness value 0.5. Then the x-axis and y-axis edges are got corresponding to the original images.

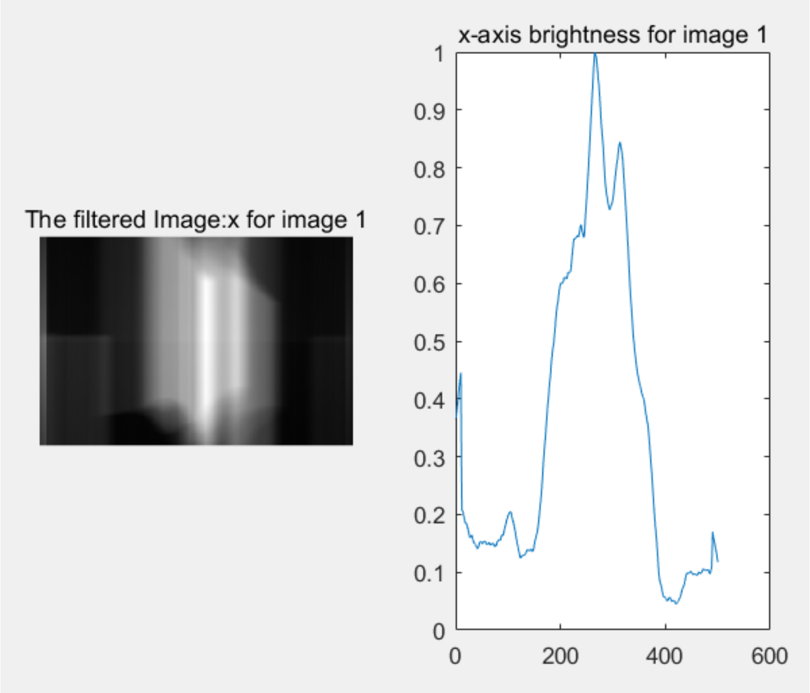

Figure 6.36. The filtered brightness result of image 1 (x-axis)

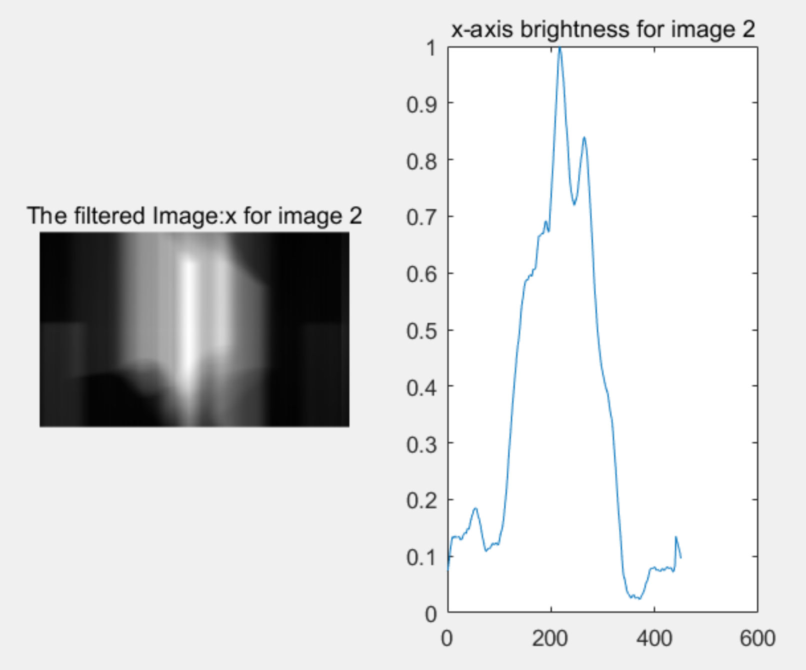

Figure 6.37. The filtered brightness result of image 2 (x-axis)

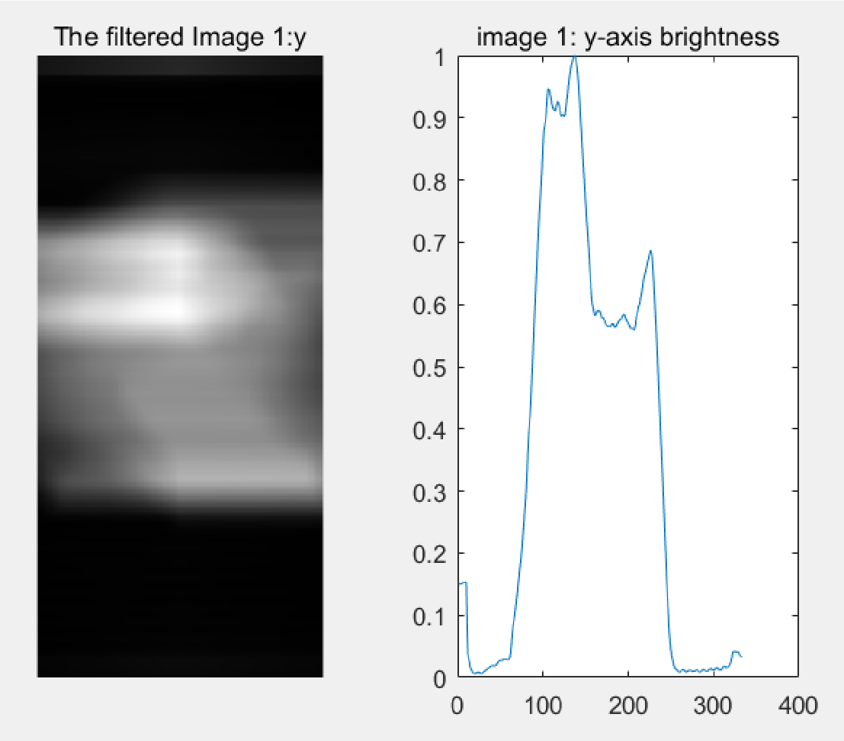

Figure 6.38. The filtered brightness result of image 1 (y-axis)

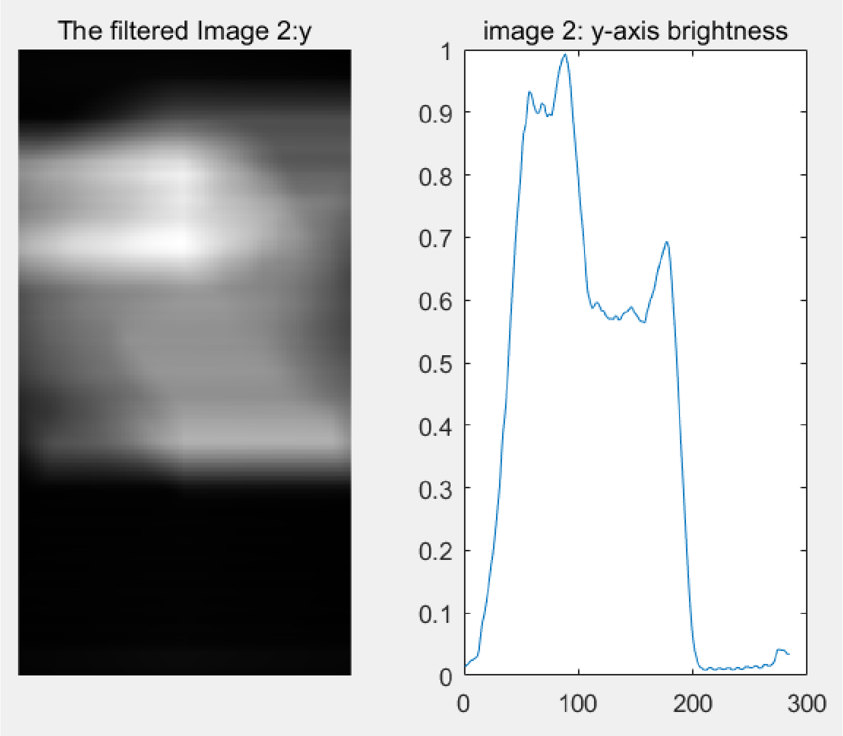

Figure 6.39. The filtered brightness result of image 2 (y-axis)

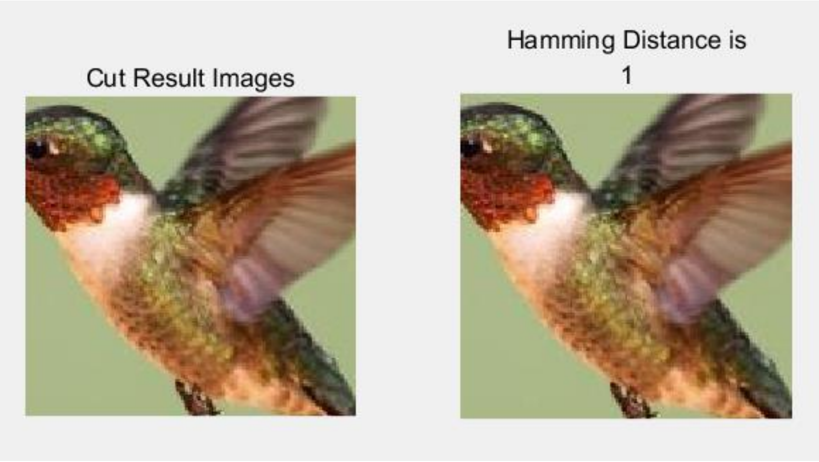

Figure 6.40. The final cut results of 2 bird images and their Hamming Distance

Inside figure 6.40 are the final results. The hamming distances for the 2 final images is only 1, which is a very satisfaction result.

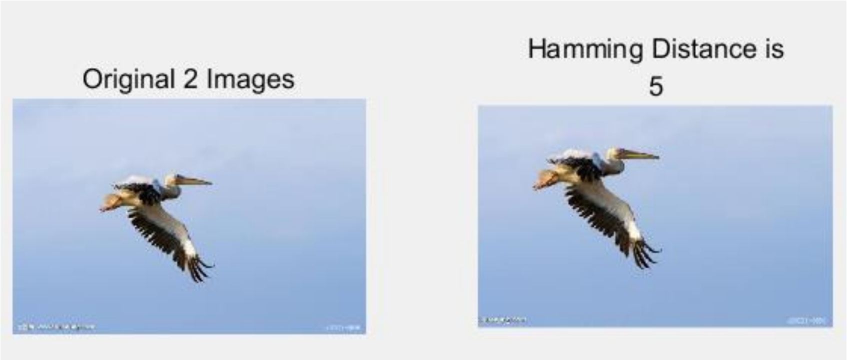

Figure 6.41. The example original bird images and there hamming distance (2)

Figures 6.41 is another example of bird and their hamming distance is 5. Figures 6.42 to 6.45 are the filtered brightness value results of the 2 birds' images in horizontal and perpendicular directions, and their one-dimension figures respectively.

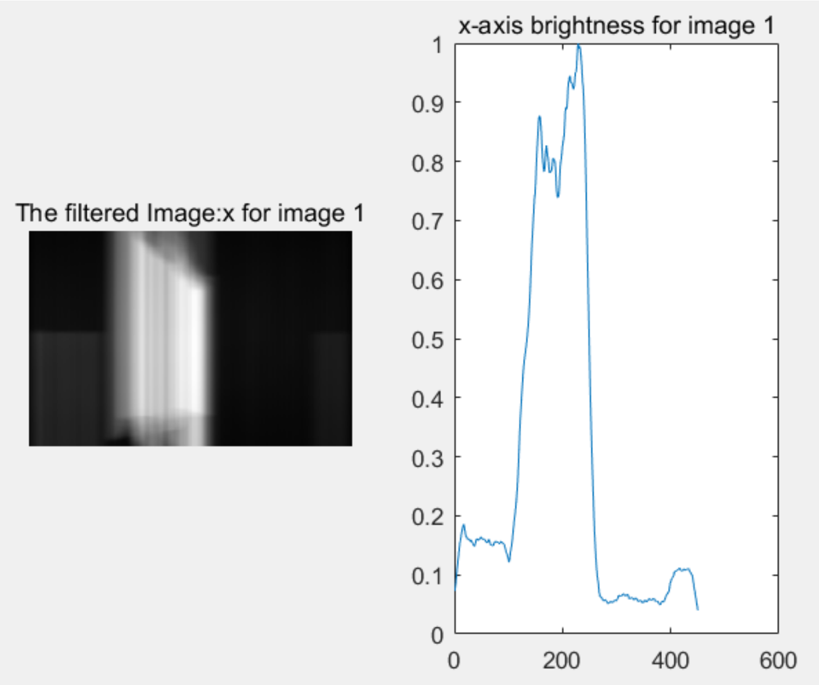

Figure 6.42. The filtered brightness result of image 1 (x-axis)

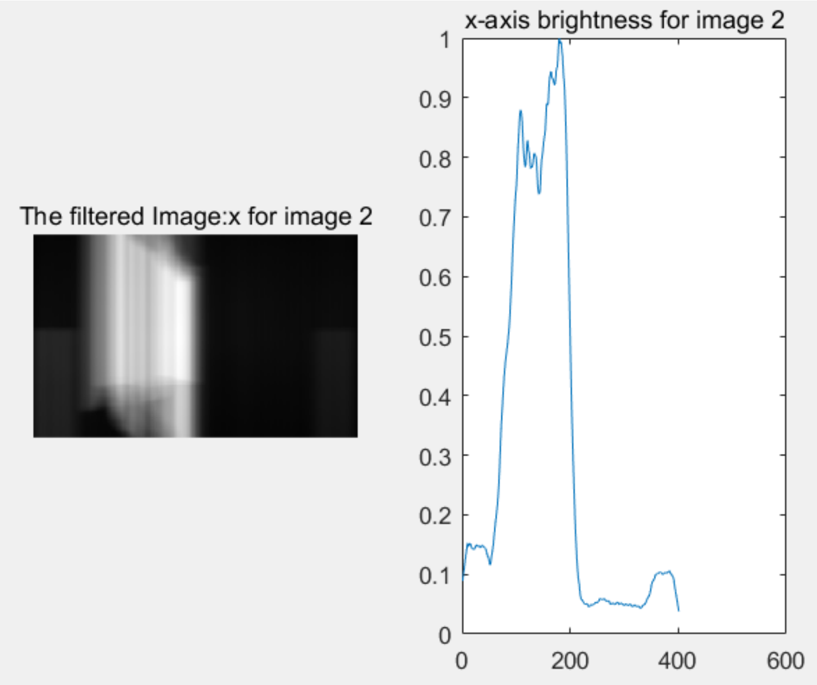

Figure 6.43. The filtered brightness result of image 2 (x-axis)

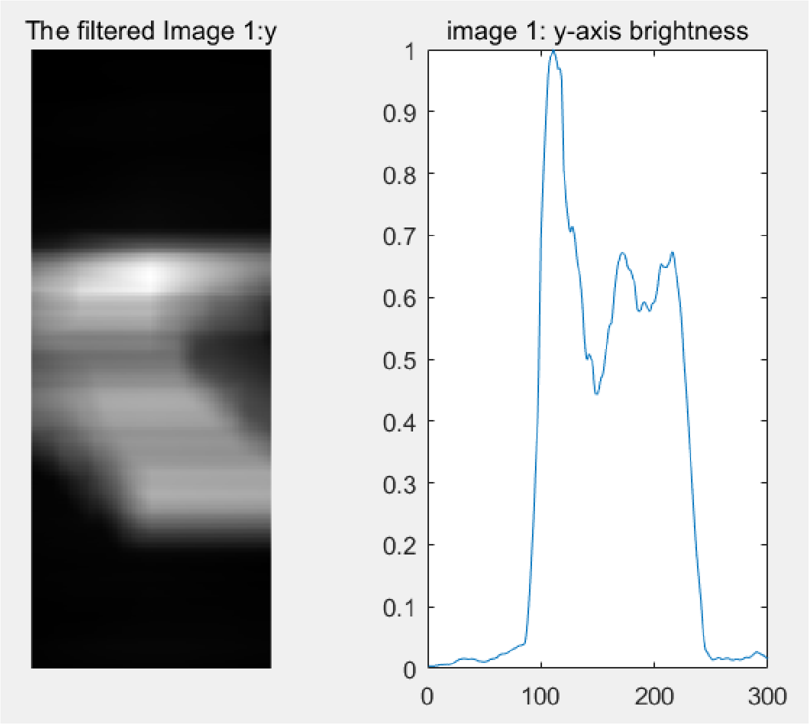

Figure 6.44. The filtered brightness result of image 1 (y-axis)

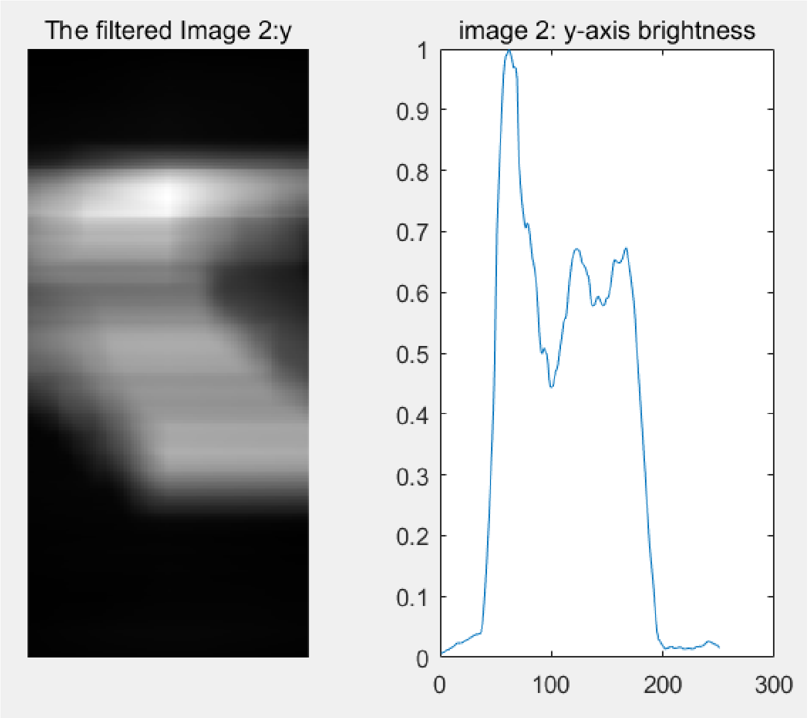

Figure 6.45. The filtered brightness result of image 2 (y-axis)

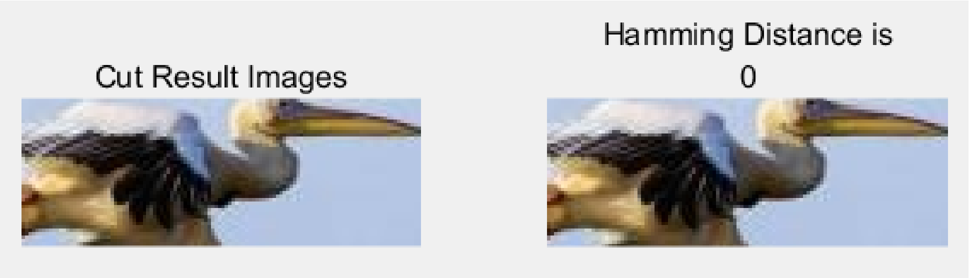

Figure 6.46. The final cut results of 2 bird images and their Hamming Distance

The final results are in figure 6.46 and again the hamming distances for the 2 final images is 0, which indicate that the original 2 images are in fact the dame bird.

6.5 Summary

In this chapter, we combine image Hash with our novel density scan method to solve the problem of retrieving near duplicate images. As far as we know, we are the first to utilize density scan method to figure out the objects within images, and the most important is that this method is simple but effective. We test our novel method with different images including cars, cats and birds. The results of the experiments indicate that our algorithm is very useful to improve image Hash methods.

CHAPTER 7

CONCLUSION

Nowadays, huge videos data is generated rapidly; hence the research of videos data is the key point of big data research areas, especially inside urban surveillance systems.

The applications of urban surveillance systems are widely distributed around us, such as counter surveillance in banks, entrance monitoring, traffic monitoring, and so on. In these systems, there are huge amount of cameras working all the time. The explosive increasing number of video resources has brought an urgent need to develop intelligent methods to reduce the pressure on the management processes.

In this thesis, we research on several topics to reduce the big data which is needed to be processes in urban surveillance systems. The topics include low-complexity intelligent big video data recording algorithms to save the HDs' space; to reduce the visual block artifacts of JPEC images for Urban Surveillance Systems; to design a novel filter to recognize vehicle plates (or vehicles) inside the images; to improve our filter to identify whether there are objects in these images captured by the cameras, hence the background subtraction is easy to be achieved; and to combine the image Hash methods with our proposed filters to retrieve similar duplicate images, which is difficult of accomplishment with only the image Hash methods.

In chapter 4, we proposed a novel filter to find out the high energy part of an image, which is useful to detect vehicles or vehicle plates from images, since the objects are commonly the highest energy concentrated area. We expand the useful area of this kind of filters in the following chapters 5 and 6. The merit of our method is the novelty of the filter, since no one else has proposed similar method to retrieve objects; but our proposed filter still has its weak point. If the environment is too complex, it may face difficulty to filter out the desired objects. In the future research work, more efforts will be needed to solve this problem.

REFERENCES

[1] D. Laney, "3D Data Management: Controlling Data Volume, Velocity, and Variety," META Group Technical Report 2001. [Online]. http://blogs.gartner.com/doug-laney/files/2012/01/ad949-3D-Data-Management-Controlling-Data-Volume-Velocity-and-Variety.pdf.

[2] http://www.cisco.com/c/en/us/solutions/collateral/service-provider/ip-ngn-ip-next-generation-network/white_paper_c11-481360.html

[3] Z. Xu, L. Mei, Y. Liu and C. Hu, "Video Structural Description: a Semantic based Model for Representing and Organizing Video Surveillance Big Data," 2013 IEEE 16th International Conference on Computational Science and Engineering, pp. 802-809.

[4] S. Chen, K. Clawson, M. Jing, J. Liu, H. Wang and B. Scotney, "Uncertainty Reasoning Based Formal Framework for Big Video Data Understanding," 2014 IEEE/WIC/ACM International Joint conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT), pp. 487-494.

[5] R. S. Feris, B. Siddiquie, J. Petterson, Z. Yun, A. Datta, L. M. Brown and S. Pankanti, "Large-Scale Vehicle Detection, Indexing, and Search in Urban Surveillance Videos," IEEE Transactions on Multimedia, Page(s): 28 – 42, 2012.

[6] S. Zhang, S. C. Chan, B. Liao and K. M. Tsui, "A New Visual Object Tracking Algorithm Using Bayesian Kalman Filter," 2014 IEEE International Symposium on Circuits and Systems (ISCAS), pp. 522-525, 2014.

[7] C. A. Segall, R. Molina, A. K. Katsaggelos and J. Mateos, “Bayesian High-resolution Reconstruction of Low-resolution Compressed Video,” IEEE International conference on Image Processing, 7-10 Oct. 2001, pp. 25-28.

[8] C. A. Segall and A. K. Katsaggelos, “Bayesian Resolution Enhancement of Compressed Video,” IEEE Transactions on Image Processing, Vol. 13, No. 7, July 2004, pp. 898-911.

[9] Y. Hwang, M. Lyu and C. Lin, “A Low Complexity Embedded Compression Codec Design with Rate Control for High Definition Video,” IEEE Transactions on Circuits and Systems for Video Technology, Vol. 25, No. 4, pp. 674-687, April 2015.

[10] J. Choi, S.W. Han, S. J. Kim, S. I. Chang and E. Yoon, “A Spatial-Temporal Multiresolution CMOS Image Sensor With Adaptive Frame Rates for Tracking the Moving Objects in Region-of-Interest and Suppressing Motion Blur,” IEEE Journal of Solid-State Circuits, Volume: 42 , Issue: 12 , pp. 2978 - 2989, 2007.

[11] R.Singh,V. Srivastava,“JPEG2000:A review and its performance comparison with JPEG,” 2012 2nd International Conference on Power, Control and Embedded Systems (ICPCES), 2012, pp. 1-7.

[12] M. Jha, A. Yumnam, D. Raju, “Comparison between image codecs: JPEG and JPEG2000, ” Computing for Sustainable Global Development (INDIACom), 2014 International Conference on, 2014, Pages: 532 - 535.

[13] E.Balster, B. Fortener, W. Turri, “Integer computation of JPEG2000 wavelet transform and quantization for lossy compression,” 2010 7th International Symposium on Communication Systems Networks and Digital Signal Processing (CSNDSP), 2010, Pages: 495 – 500.

[14] Bo Li, Bin Tian, Qingming Yao, Kunfeng Wang, "A vehicle license plate recognition system based on analysis of maximally stable extremal regions," 2012 9th IEEE International Conference on Networking Sensing and Control (ICNSC), Pages: 399 - 404, 2012.

[15] Anagnostopoulos, C.-N.E.; Anagnostopoulos, I.E.; Psoroulas, I.D.; Loumos, V.;Kayafas, E., "License Plate Recognition From Still Images and Video Sequences: A Survey," IEEE Transactions on Intelligent Transportation Systems, Volume: 9, Issue: 3, Pages: 377 - 391, 2008.

[16] Mei Yu; Yong Deak Kim, "An approach to Korean license plate recognition based on vertical edge matching," 2000 IEEE International Conference on Systems, Man, and Cybernetics, Volume: 4, Pages: 2975 – 2980, 2000.

[17] Al-Ghaili, A.M.; Mashohor, S.; Ramli, A.R.; Ismail, A., "Vertical-Edge-Based Car-License-Plate Detection Method," IEEE Transactions on Vehicular Technology, Volume: 62, Issue: 1, Pages: 26 - 38, Year: 2013

[18] Hsieh, Jun-Wei; Shih-Hao Yu; Yung-Sheng Chen, "Morphology-based license plate detection from complex scenes," Pattern Recognition, 2002.16th International Conference on Proceedings. Pages: 176 – 179, Volume: 3, Year: 2002.

[19] Lensky, Artem A.; Kang-Hyun Jo; Gubarev, Vasiliy V., "Vehicle License Plate Detection using Local Fractal Dimension and Morphological Analysis," The 1st International Forum on Strategic Technology, Pages: 47 - 50, Year: 2006.

[20] Shen-Zheng Wang; Hsi-Jian Lee, "Detection and recognition of license plate characters with different

appearances," Proceedings of the 2003 IEEE Intelligent Transportation Systems,Pages: 979 - 984 Volume: 2, Year: 2003.

[21] Bai Hongliang; Liu Changping, "A hybrid license plate extraction method based on edge statistics and morphology," Proceedings of the 17th International Conference on Pattern Recognition. Pages: 831 - 834 Volume: 2, Year: 2004.

[22] Gang Li; Rongdi Yuan; Zuyuan Yang; Xiyue Huang, "A Yellow License Plate Location Method Based on RGB Model of Color Image and Texture of Plate," Second Workshop on Digital Media and its Application in Museum & Heritages, Pages: 42 - 46, Year: 2007.

[23] Ahmadyfard, Alireza ; Abolghasemi, Vahid, "Detecting license plate using texture and color information," International Symposium on Telecommunications, Pages: 804 - 808, Year: 2008.

[24] Lin Luo; Hao Sun; Weiping Zhou; Limin Luo, "An Efficient Method of License Plate Location," 2009 1st International Conference on Information Science and Engineering (ICISE), Pages: 770 - 773, Year: 2009.

[25] Guangmin Sun; Canhui Zhang; Weiwei Zou; Guangyu Yu, "A new recognition method of vehicle license plate based on Genetic Neural Network," 2010 the 5th IEEE Conference on Industrial Electronics and Applications (ICIEA), Pages: 1662 – 1666, Year: 2010.

[26] YiQing Liu; Dong Wei; Ning Zhang; MinZhe Zhao, "Vehicle-license-plate recognition based on neural networks," 2011 IEEE International Conference on Information and Automation (ICIA), Pages: 363 - 366, Year: 2011.

[27] Sharma, J.; Mishra, A.; Saxena, K.; Kumar, S., "A hybrid technique for License Plate Recognition based on feature selection of wavelet transform and artificial neural network," Technology (ICROIT), 2014 International Conference on Optimization, Reliability, and Information. Pages: 347 – 352, Year: 2014.

[28] Rajput, Hitesh; Som, Tanmoy; Kar, Soumitra, "An Automated Vehicle License Plate Recognition System," Computer, Volume: 48, Issue: 8, Pages: 56 - 61, Year: 2015.

[29] J. Zhang and M. L. Huang, "5Ws Model for Big Data Analysis and Visualization", 2013 IEEE 16th International Conference on Computational Science and Engineering, pp. 1021-1028.

[30] Cisco Visual Networking Index: Forecast and Methodology, 2014-2019 White Paper http://www.cisco.com/c/en/us/solutions/collateral/service-provider/ip-ngn-ip-next-generation-network/white_paper_c11-481360.html.

[31] Z. Xu, L. Mei, Y. Liu and C. Hu, "Video Structural Description: a Semantic based Model for Representing and Organizing Video Surveillance Big Data", 2013 IEEE 16th International Conference on Computational Science and Engineering, pp. 802-809.

[32] S. Chen, K. Clawson, M. Jing, J. Liu, H. Wang and B. Scotney, "Uncertainty Reasoning Based Formal Framework for Big Video Data Understanding", 2014 IEEE/WIC/ACM International Joint conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT), pp. 487-494.

[33] R. S. Feris, B. Siddiquie, J. Petterson, Z. Yun, A. Datta, L. M. Brown and S. Pankanti, "Large-Scale Vehicle Detection, Indexing, and Search in Urban

Surveillance Videos", IEEE Transactions on Multimedia, Page(s): 28 -- 42, 2012.

[34] S. Zhang, S. C. Chan, B. Liao and K. M. Tsui, "A New Visual Object Tracking Algorithm Using Bayesian Kalman Filter", 2014 IEEE International Symposium on Circuits and Systems (ISCAS), pp. 522-525, 2014.

[35] C. A. Segall, R. Molina, A. K. Katsaggelos and J. Mateos, "Bayesian High-resolution Reconstruction of Low-resolution Compressed Video", IEEE International conference on Image Processing, 7-10 Oct. 2001, pp. 25-28.

[36] C. A. Segall and A. K. Katsaggelos, "Bayesian Resolution Enhancement of Compressed Video", IEEE Transactions on Image Processing, Vol. 13, No. 7, July 2004, pp. 898-911.

[37] Y. T. Hwang, M. W. Lyu and C. C. Lin, "A Low Complexity Embedded Compression Codec Design with Rate Control for High Definition Video", IEEE Transactions on Circuits and Systems for Video Technology, Vol. 25, No. 4, pp. 674-687, April 2015.

[38] J. Choi, S.W. Han, S. J. Kim, S. I. Chang and E. Yoon, "A Spatial-Temporal Multiresolution CMOS Image Sensor With Adaptive Frame Rates for Tracking the Moving Objects in Region-of-Interest and Suppressing Motion Blur", IEEE Journal of Solid-State Circuits, Volume: 42, Issue: 12 , Page(s): 2978 - 2989, 2007.

[39] W. H. Zhang, H. G. Xu, B. Wu and S. Q. Li, "Safety Management of Traffic Accident Scene Based on System Dynamics", 2008 International Conference on Intelligent Computation Technology and Automation (ICICTA), Page(s): 482 -- 485, 2008.

[40] R. Singh and V.K. Srivastava, “JPEG2000: A review and its performance comparison with JPEG”. Power, Control and Embedded Systems (ICPCES), 2012 2nd International Conference on, 2012, Pages: 1-7.

[41] M. Jha, A.S.Yumnam, D. Raju, “Comparison between image codecs: JPEG and JPEG2000”, Computing for Sustainable Global Development (INDIACom), 2014 International Conference on, 2014, Pages: 532 - 535.

[42] S. Vishaga and S. L. Das, “A survey on switching median filters for impulse noise removal”, Power and Computing Technologies (ICCPCT), 2015 International Conference on Circuit, 2015, Pages: 1 - 6.

[43] A. N. Venetsanopoulos and K. N. Plataniotis, “Adaptive filters for color image processing: A survey”, Signal Processing Conference, 2000 10th European, 2000, Pages: 1 - 4.

[44] G. A. Triantafyllidis, D. Tzovaras, D. Sampson, M. G. Strintzis, “A hybrid algorithm for the removal of blocking artifacts”, ICME '02. Proceedings. 2002 IEEE International Conference on Multimedia and Expo, 2002. Volume: 1, Pages: 161 - 164.

[45] Y. Kwon, K. I. Kim, J. Tompkin, J. H. Kim, C. Theobalt, “Efficient Learning of Image Super-Resolution and Compression Artifact Removal with Semi-Local Gaussian Processes”, IEEE Transactions on Pattern Analysis and Machine Intelligence, 2015, Volume: 37, Issue: 9, Pages: 1792 - 1805.

[46] C. Tang, A. W. K. Kong, N. Craft, “A knowledge-based Algorithm to remove blocking artifacts in skin images for forensic analysis”, 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2011, Pages: 1928 - 1931.

[47] Y. Luo, R. K. Ward, “Removing the blocking artifacts of block based DCT compressed images”, IEEE Transactions on Image Processing, 2003, Volume: 12, Issue: 7, Pages: 838 - 842.

[48] J. Chou, M. Crouse, K. Ramchandran, “A simple algorithm for removing blocking artifacts in block-transform coded images”, IEEE Signal Processing Letters, 1998, Volume: 5, Issue: 2, Pages: 33 - 35.

[49] S. Chen, H. Xu, D. Liu, B. Hu, H. Wang, “A vision of IoT: applications, challenges, and opportunities with China perspective,” IEEE Internet of Things Journal, Vol. 1, Issue 4, pp. 349 - 359, Aug 2014.

[50] M. Handte, S. Foell, S. Wagner, G. Kortuem, P. Marrón; “An Internet-of-Things enabled connected navigation system for urban bus riders,” IEEE Internet of Things Journal, Vol. 3, Issue 5, pp. 735 - 744, Oct 2016.

[51] B. Li, B. Tian, Q. Yao, K. Wang, “A vehicle license plate recognition system based on analysis of maximally stable extremal regions,” 9th IEEE International Conference on Networking, Sensing and Control, pp. 399 - 404, 2012.

[52] C. Anagnostopoulos, I. Anagnostopoulos, I. Psoroulas, V. Loumos, E. Kayafas, “License plate recognition from still images and video sequences: a survey,” IEEE Transactions on Intelligent Transportation Systems, Vol. 9, Issue 3, pp. 377 - 391, Sept 2008.

[53] M. Yu, Y. Kim, “An approach to Korean license plate recognition based on vertical edge matching,” 2000 IEEE International Conference on Systems, Man, and Cybernetics, Vol. 4, pp. 2975 – 2980, 2000.

[54] A. Al-Ghaili, S. Mashohor, A. Ramli, A. Ismail, “Vertical-edge-based car-license-plate detection method,” IEEE Transactions on Vehicular Technology, Vol. 62, Issue 1, pp. 26 - 38, Jan. 2013.

[55] J. Hsieh, S. Yu, Y. Chen, “Morphology-based license plate detection from complex scenes,” 16th IEEE International Conference on Pattern Recognition, pp. 176 – 179, 2002.

[56] A. Lensky, K. Jo, V. Gubarev, “Vehicle license plate detection using local fractal dimension and morphological analysis,” The 1st IEEE International Forum on Strategic Technology, pp. 47 - 50, 2006.

[57] S. Wang, H. Lee, “Detection and recognition of license plate characters with different appearances,” Proceedings of IEEE Intelligent Transportation Systems, pp. 979 – 984, Vol 2, 2003.

[58] H. Bai, C. Liu, “A hybrid license plate extraction method based on edge statistics and morphology,” 17th IEEE International Conference on Pattern Recognition, pp. 831 – 834, Vol. 2, 2004.

[59] G. Li, R. Yuan, Z. Yang, X. Huang, “A yellow license plate location method based on RGB model of color image and texture of plate,” Second Workshop on Digital Media and its Application in Museum & Heritages, pp. 42 - 46, 2007.

[60] Ahmadyfard, V. Abolghasemi, “Detecting license plate using texture and color information,” International Symposium on Telecommunications, pp. 804 - 808, 2008.

[61] L. Luo, H. Sun, W. Zhou, L. Luo, “An efficient method of license plate location,” 1st International Conference on Information Science and Engineering, pp. 770 - 773, 2009.

[62] G. Sun, C. Zhang, W. Zou, G. Yu, “A new recognition method of vehicle license plate based on genetic neural network,” The 5th IEEE Conference on Industrial Electronics and Applications, pp. 1662 – 1666, 2010.

[63] Y. Liu, D. Wei, N. Zhang, M. Zhao, “Vehicle-license-plate recognition based on neural networks,” IEEE International Conference on Information and Automation, pp. 363 - 366, 2011.

[64] J. Sharma, A. Mishra, K. Saxena, S. Kumar, “A hybrid technique for license plate recognition based on feature selection of wavelet transform and artificial neural network,” International Conference on Reliability, Optimization and Information Technology, pp. 347 – 352, 2014.

[65] H. Rajput, T. Som, S. Kar, “An automated vehicle license plate recognition system,” Computer, Vol. 48, Issue 8, pp. 56 - 61, Aug 2015.

[66] O. Mendoza-Schrock, M. Rizki, V. Velten, “Manifold and transfer subspace learning for cross-domain vehicle recognition in dynamic systems,” 18th International Conference on Information Fusion, pp. 1954 – 1961, 2015.

[67] W. Pei, Z. An, Y. Zhu, X. Jia, X. Zuo, F. Wang, “A rapid vehicle recognition and retrieval system,” 2nd International Conference on Systems and Informatics, pp. 748 - 753, 2014.

[68] G. Burresi, R. Giorgi, “A field experience for a vehicle recognition system using magnetic sensors,” The 4th Mediterranean Conference on Embedded Computing, pp. 178 – 181, 2015.

[69] M. V. Moreno, F. Terroso-Sáenz, Aurora González-Vidal, M. Valdés-Vela, Antonio F. Skarmeta, Miguel A. Zamora, Victor Chang, “Applicability of Big Data Techniques to Smart Cities Deployments”, IEEE

Transactions on Industrial Informatics, Vol. 13, Issue 2, pp. 800 – 809, April 2017.

[70] M. Mazhar Rathore, Awais Ahmad, Anand Paul, “IoT-based Smart City Development using Big Data Analytical Approach”, 2016 IEEE International Conference on Automatica (ICA-ACCA), pp. 1-8, 2016.

[71] S. Chen, H. Xu, D. Liu, B. Hu, H. Wang, “A Vision of IoT: Applications, Challenges, and Opportunities with China Perspective,” IEEE Internet of Things Journal, Vol. 1, Issue 4, pp. 349 - 359, Aug 2014.

[72] M. Handte, S. Foell, S. Wagner, G. Kortuem, P. Marrón, “An Internet-of-Things Enabled Connected Navigation System for Urban Bus Riders,” IEEE Internet of Things Journal, Vol. 3, Issue 5, pp. 735 - 744, Oct 2016.

[73] Honghai Liu, Xianghua Hou, “Moving Detection Research of Background Frame Difference Based on Gaussian Model”, 2012 International Conference on Computer Science and Service System, pp258 – 261, Year: 2012.

[74] X. Han, Y. Gao, Z. Lu, “Research on Moving Object Detection Algorithm Based on Improved Three Frame Difference Method and Optical Flow”, Fifth International Conference on Instrumentation and Measurement, Computer, Communication and Control, 2015.

[75] Y. Zhuang, C. Wu, Y. Zhang, S. Feng, “Detection and Tracking Algorithm based on Frame Difference Method and Particle Filter Algorithm”, 2017 29th Chinese Control And Decision Conference (CCDC), pp. 161 – 166.

[76] J. Suhr, H. Jung, G. Li, J. Kim, “Mixture of Gaussians-Based Background Subtraction for Bayer-Pattern Image Sequences”, IEEE

Transactions on Circuits and Systems for Video Technology, pp. 365 – 370, Vol. 21, Issue 3, March 2011.

[77] Dibyendu Mukherjee, Q. M. Jonathan Wu, Thanh Minh Nguyen, "Gaussian Mixture Model With Advanced Distance Measure Based on Supp ort Weights and Histogram of Gradients for Background Suppression", IEEE Transactions on Industrial Informatics, pp. 1086 – 1096, Vol. 10, Issue 2, May 2014.

[78] V. Sikri, "Proposition and Comprehensive Efficiency Evaluation of a Foreground Detection Algorithm based on Optical Flow and Canny Edge Detection for Video Surveillance Systems", 2016 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET), pp. 1466 – 1472, 2016.

[79] V. Sikri, "Proposition and Comprehensive Efficiency Evaluation of a Foreground Detection Algorithm based on Optical Flow and Canny Edge Detection for Video Surveillance Systems", 2016 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET), pp. 1466 -1472, 2016.

[80] L. Hu, Q. Ni, "IoT-Dreven Automated Object Detection Algorithm for Urban Surveillance Systems in Smart Cities", IEEE Internet of Things Journal, Volume: 5, Issue: 2, Pages: 747 - 754, 2018.

[81] Yuan Cao; Heng Qi; Jien Kato; Keqiu Li, "Hash Ranking With Weighted Asymmetric Distance for Image Search", IEEE Transactions on Computational Imaging, Volume: 3, Issue: 4, Pages: 1008 – 1019, 2017.

[82] Mayank Srivastava; Jamshed Siddiqui; Mohammad Athar Ali, "Robust image hashing based on statistical features for copy detection", 2016 IEEE

Uttar Pradesh Section International Conference on Electrical, Computer and Electronics Engineering (UPCON), Pages: 490 – 495, 2016.

[83] Satendra Pal Singh; Gaurav Bhatnagar, A robust image hashing based on discrete wavelet transform", 2017 IEEE International Conference on Signal and Image Processing Applications (ICSIPA), Pages: 440 – 444, 2017. "

[84] Cai-Ping Yan; Chi-Man Pun; Xiao-Chen Yuan, "Quaternion-Based Image Hashing for Adaptive Tampering Localization", IEEE Transactions on Information Forensics and Security, Volume: 11, Issue: 12, Pages: 2664 – 2677, 2016.

[85] Cai-Ping Yan; Chi-Man Pun; Xiao-Chen Yuan, "Adaptive local feature based multi-scale image hashing for robust tampering detection", TENCON 2015 - 2015 IEEE Region 10 Conference, Pages: 1 – 4, 2015.

[86] Yan Zhao; Shuozhong Wang; Xinpeng Zhang; Heng Yao, "Robust Hashing for Image Authentication Using Zernike Moments and Local Features", IEEE Transactions on Information Forensics and Security, Volume: 8, Issue: 1, Pages: 55 – 63, 2013.

[87] Xiaofeng Wang; Kemu Pang; Xiaorui Zhou; Yang Zhou; Lu Li; Jianru Xue, "A Visual Model-Based Perceptual Image Hash for Content Authentication", IEEE Transactions on Information Forensics and Security, Volume: 10, Issue: 7, Pages: 1336 – 1349, 2015.

[88] Stas Goferman; Lihi Zelnik-Manor; Ayellet Tal, "Context-Aware Saliency Detection", IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume: 34, Issue: 10, Pages: 1915 – 1926, 2012.

[89] Tie Liu; Zejian Yuan; Jian Sun; Jingdong Wang; Nanning Zheng; Xiaoou Tang; Heung-Yeung Shum, Learning to Detect a Salient Object", IEEE

Transactions on Pattern Analysis and Machine Intelligence, Volume: 33, Issue: 2, Pages: 353 – 367, 2011.

[90] Ali Borji; Ming-Ming Cheng; Huaizu Jiang; Jia Li, “Salient Object Detection: A Benchmark”, IEEE Transactions on Image Processing, Volume: 24, Issue: 12, Pages: 5706 – 5722, 2015.