Machine Learning Methods for Autonomous Flame Detection in Videos

Zhenglin Li

Department of Automatic Control and Systems Engineering
University of Sheffield

This dissertation is submitted for the degree of
Doctor of Philosophy

June 2020

Acknowledgements

I would like to express my great appreciation to my supervisor Prof. Lyudmila Mihaylova for her kind help on both my research and life. She is always energetic and ready to help her students in their research. Her inspiring ideas and advice showed me the direction of research.

I am grateful to Dr Dana Damian who gave me great help and valuable advice on the career planning. Really appreciated Dr Le Yang for his support and helpful comments on my papers. His hard-working and enthusiasm of research impressed me and changed my way of working. I would like to thank my collaborator Dr Lucile Rossi as well for their data.

Great thanks also go to my colleague Dr Olga Isupova, who is nice and ready to help all the time. I learned a lot from the discussion with her on my PhD research. Our collaboration in my first year provided me with clear guidance on the way of performing research. Big thanks to all the nice colleagues of Prof. Lyudmila Mihaylova's group. It is really lucky to be a member of you. Your support made my PhD a great journey.

I would like to acknowledge the China Scholarship Council and the Department of Automatic Control and Systems Engineering for the funding which enables me to start my PhD study.

Great thanks to my mum Prof. Yufen Li, my boyfriend and all my family. Your supports and comforts from thousands of miles away mean a lot to me and helped me to get through the difficult time.

I am also grateful to my friends Ke Sun, Yaxin Li, Jinny Robinson, Yang Zhang, Will Jacobs, Uziel Avila and every friend I met in the UK. Thank you so much for your company and supports.

Abstract

Fire detection has attracted increasing attention from the public because of the huge loss caused by fires every year. Compared with the traditional fire detection techniques based on smoke or heat sensors, the frameworks using machine learning methods in videos for fire detection have the advantages of higher efficiency and accuracy of detection, robustness to various environments, and lower cost of the systems. The uniqueness of these frameworks stems from the developed machine learning approaches for autonomous information extraction and fire detection in sequential video frames.

A framework for flame detection is proposed in Chapter 3, based on the synergy of the Horn-Schunck optical flow estimation method, a probabilistic saliency analysis approach and a temporal wavelet analysis scheme. The estimated optical flows, together with the saliency analysis method, work effectively in selecting moving regions by well describing the dynamic property of flames, which contributes to accurate detection of flames. Additionally, the temporal wavelet transform based analysis increases the robustness of the framework and provides reliable results by discarding non-flame pixels according to their temporally changing patterns.

Apart from the dynamic characteristic of flames, the property of colours is also of crucial importance in describing flames. However, the colours of flames usually vary significantly with different illumination or burning material, which results in a wide diversity. To well model the various colours, a novel flame colour model is proposed in Chapter 4 based on the Dirichlet process Gaussian mixture model. The distribution of flame colours is represented by a Gaussian mixture model, of which the number of mixture components is learned from the training data autonomously by setting a Dirichlet process as the prior. Compared with those methods which set the number of mixture components empirically, the developed model can access a more accurate estimation of the distribution of flame colours. The inference is successfully implemented by two methods, i.e., the Gibbs sampling and variational inference algorithms, to manage different quantities of training data. The colour model can be incorporated into the framework of flame detection proposed in Chapter 3, and the results show that the colour model achieves a highly accurate estimation of the distribution of flame colours, which contributes to the good performance of the whole framework.

In Chapter 5, two frameworks for flame detection are developed based on the flame region-based convolutional neural network and the faster region-based convolutional neural network, respectively. To overcome the difficulties in flame detection caused by the remarkably diverse appearance of flames, a novel flame proposal generation scheme is proposed in the former framework, based on the combination of the flame colour model and online robust principal component analysis algorithm. It works effectively in generating proposals containing flames by taking the dynamic and colour properties of flames into consideration. In the latter framework using the faster region-based convolutional network, the flame proposals are generated by a region proposal generation network. In both frameworks, the generated proposals are subsequently projected onto a feature map produced by several convolutional layers and are further processed by additional layers. Regions of flames are outputted by both frameworks, based on which frame-wise results can be obtained.

All the proposed approaches are tested on real videos of various environments and proved to be capable of accurate flame detection.

Contents

List of Figures

List of Tables

Glossary

Adam	adaptive moment estimation
CNN	convolutional neural network
CRP	Chinese restaurant process
DP	Dirichlet process
DPGMM	Dirichlet process Gaussian mixture model
ELBO	evidence lower bound
Fast-VDP	accelerated variational Dirichlet process
FFA	fuzzy finite automata
FPN	feature pyramid network
FPR	false positive rate
FWT	fast wavelet transform
GAN	generative adversarial network
GMM	Gaussian mixture model
GS	Gibbs sampling
IoU	intersection over union
KL	Kullback-Leibler
MCMC	Markov chain Monte Carlo
NSD	non-smooth data
OMT	optimal mass transport

OR-PCA	online robust principal component analysis
PCP	principal component pursuit
R-CNN	region based CNN
ReLU	rectified linear unit
ROC	receiver operating characteristic
RoI	region of interest
RPCA	robust principal component analysis
SVM	support vector machine
TNR	true negative rate
TPR	true positive rate
VI	variational inference
YOLO	you only look once

Nomenclature

In this thesis, matrices are denoted with uppercase bold letters, e.g., $\mathbf{M}$, and vectors are denoted with lowercase bold letters, e.g., $\mathbf{m}$. The superscript $\top$ stands for transposition. For matrix $\mathbf{M}$, $\mathrm{tr}(\mathbf{M})$ denotes its trace and $\mathrm{rank}(\mathbf{M})$ denotes its rank. The Frobenius norm is denoted by $\|\mathbf{M}\|_F$ and $\|\mathbf{M}\|_*$ denotes the nuclear norm of the matrix $\mathbf{M}$.

Roman Symbols

$Beta(\cdot)$	Beta distribution
d	Dimension of a vector
$\det(\cdot)$	Determinant of a matrix
$\widetilde{F}(\cdot)$	Likelihood function
$F(x_s, y_s)$	Feature of the pixel located at (x_s, y_s)
$f_n(\cdot)$	Empirical cost function with n samples
G	Prior
G_0	Base distribution of the Dirichlet Process
$g_t(\cdot)$	Surrogate function with t samples of the empirical cost function $f_n(\cdot)$
I_d	Identity matrix of dimension d
k^*	Index indicating a new cluster
K_l	Number of Latent variables
$\mathbf{L}$	Matrix of the basis of the low dimensional subspace
$\mathbf{M}$	Matrix of observed vectorized gray-scale frames of videos
$\mathbf{m}_i$	The i-th observed vectorized gray-scale frame in a video
$\mathcal{N}(\cdot)$	Gaussian distribution

N	Number of data cases
$\mathcal{Q}$	A family of variational distributions
$q(\cdot)$	Variational distribution
$q^*(\cdot)$	Optimal distribution among the variational distribution family $\mathcal{Q}$
$\mathbf{R}$	Coefficients of samples projected to the basis of low dimensional subspace $\mathbf{L}$
r	Upper bound of the rank of the low rank matrix $\mathbf{Y}$
$\mathbf{S}$	Sparse matrix
t	Temporal index of a frame, discrete variable
t_c	Time, continuous variable
(u,v)	Optical flow vector
$\dot{V}_t$	Partial derivative of the intensity V with respect to time t_c
$\dot{V}_x$	Partial derivative of the intensity V with respect to spatial abscissa x_c
$\dot{V}_y$	Partial derivative of the intensity V with respect to spatial ordinate y_c
V	Intensity of pixels
V_B	Intensity of the blue channel in the RGB colour space
V_G	Intensity of the green channel in the RGB colour space
V_R	Intensity of the red channel in the RGB colour space
$\mathbf{W}$	The collection of latent variables of the Dirichlet process mixture model
$\mathbf{w}$	Set of latent variables in the framework of Bayesian inference
$\mathbf{X}$	Set of observations $\mathbf{x}$
$\mathbf{x}_i$	The i-th observation in the framework of Bayesian inference
$\mathbf{X}_{-i}$	All training data except the i-th observation $\mathbf{x}_i$
x_c	Spatial abscissa along x-axis, continuous variable
x_s	Column index of a pixel in a frame, discrete variable
$\mathbf{Y}$	Low rank matrix
y_c	Spatial ordinate along y-axis, continuous variable

y_s	Row index of a pixel in a frame, discrete variable
$\mathbf{z}_{-i}$	Indicator variables of all training data except the i-th observation
z_i	Indicator variable of the i-th observation $\mathbf{x}_i$

Greek Symbols

$\mathbf{\Sigma}$	Covariance matrix of a Gaussian distribution
$\boldsymbol{\mu}$	Mean of a Gaussian distribution
λ	Hyperparameter of the base distribution G_0 of a Dirichlet process
α_0	Concentration parameter of a Dirichlet process
$\boldsymbol{\theta}_k^*$	Parameter vector specifying the k-th distinct cluster
$\boldsymbol{\theta}_i$	Parameter vector specifying the cluster associated with the observation $\mathbf{x}_i$
ϕ_k^β	Variational parameter of $q(\beta_k)$, which is part of the factorized variational distribution
$\phi_k^{\theta^*}$	Variational parameter of $q(\boldsymbol{\theta}_k^*)$, which is part of the factorized variational distribution
$\zeta(x_s, y_s)$	Saliency value of the pixel located at (x_s, y_s)

Chapter 1

Introduction

1.1 Background and Motivation

Fires cause great property damage and human casualties every year. The total losses in the USA due to fires are 329 billion USD in 2011 [6], including both direct and indirect loss. In England, there were roughly 564,000 incidents caused by fires in 2017/18 according to the fire statistical dataset published by the UK government [7]. Fires affect thousands of lives every year and make even more people homeless. Additionally, fires, especially wildfires, are among the main reasons for air pollution and global warming. The huge losses resulted from fires make fire detection techniques significantly important to the development of modern society and daily life.

The accuracy of detection is critical in the tasks of fire detection. On one hand, failing to detect fires will lead to heavy losses, which should be avoided. On the other hand, constant false alarms will disturb daily life and industrial production, and cause people to pay little attention to fire alarms, which could be a hidden danger in public safety. Additionally, timely detection helps to reduce the injuries and losses caused by fires significantly. A fire can become life-threatening in just two minutes and engulf a residence in fewer than five minutes [8]. Timely fire alarms can leave more time for people to evacuate to safety and allow firemen to start the suppression before the fires are out of control. Therefore, enhancing the accuracy and speed of detection is among the most important goals of fire detection techniques.

Manual monitoring and smoke or heat sensors were commonly employed for fire detection in the past century, but both of them have their limitations. Manual monitoring requires a large amount of labour to keep an eye on the places where fires may happen, which is a heavy burden for the governments or employers. Furthermore, most humans cannot concentrate consistently, and their attention usually degrades over time [9, 10]. The lack of concentration during surveillance may result in delayed or missed detection of fires.

Apart from manual monitoring, early research also employed sensors of smoke or heat to detect fires, by sampling soot particles, product gas, or temperatures [11]. Those sensors have been put into wide use in some developed countries. However, those techniques have many limitations as well. For example, they are confined to enclosed environments and suffer a dramatic reduction in the capability of detection when applied to large spaces, such as outdoor scenes. The degeneration of the detection performance in outdoor conditions is mainly caused by the low concentration of gas or soot particles when spreading in the air of large spaces. Additionally, the detection performance of the techniques based on smoke or heat sensors can be significantly influenced by environments, especially wind. In addition, these methods sometimes do not react quickly enough to fires, since it takes time for the smoke or gas to reach the detectors. The time of detection increases with the distances between fires and the nearest sensors. Furthermore, the high cost of the installation and maintenance of sensors limits their applications as well. In a nutshell, the fire detection approaches based on smoke or heat sensors cannot satisfy the requirements of automatic detection of fires under various circumstances.

To overcome the disadvantages of those techniques discussed in the preceding text, researchers started to work on frameworks using machine learning and computer vision methods in the past decades. These approaches extract information from videos captured by cameras to analyse the existence of fires. The detection system will send an alarm to supervising officers or even trigger an automatic fire extinguisher when a fire is detected. These methods have four major advantages over conventional techniques: first, they are capable of detecting fires accurately in large geographical areas, implying that they can be used in both indoor and wildfire detection. Next, the video-based frameworks are almost insusceptible to environmental changes. Specifically, weather changes, such as wind or rain, have a minor influence on the accuracy of detection. Third, these approaches can provide quicker and more accurate solutions [12, 13], compared with the possible long time taken by those conventional techniques discussed above. Finally, the newly developed frameworks based on computer vision methods can be easily incorporated into the existing monitoring systems without high extra costs. All the reasons mentioned above have contributed to the increasing popularity of video-based methods for fire detection recently [10].

To be put into practical application, the frameworks of fire detection should satisfy the following criteria [14]:

- **High detection accuracy**
  It is a prerequisite for the practical application of autonomous flame detection frameworks. High true positive rate is necessary because failures to detect fires will result in severe losses. Accurate detection can be challenging when fires

happen far from surveillance cameras and occupy only small regions of the scene.

- **Low false alarm rate**
  False alarm rate should be kept at a low level as well. Excessive false alarms will significantly disturb the life and work of people. However, real environments are usually complex with numerous interference, such as flashing lights and moving flame-coloured cars. Fire detection schemes also aim at eliminating the interference from non-fire objects, and thereby reduce the false alarm rate to an acceptable level.

- **Real-time processing**
  Fires usually spread rapidly where flammable materials are around, for example, in buildings or forests. Therefore, timely detection of fires will reduce the loss effectively. However, robust algorithms are sometimes computationally expensive. The accuracy and speed should be balanced to obtain optimal performance.

The methods satisfying those criteria mentioned above can be employed in practice for automatic fire surveillance to reduce the load on human operators.

The research on fire recognition based on videos is generally divided into two major branches, i.e. flame detection and smoke detection. Flame detection approaches are mostly employed in indoor environments because flames of indoor fires are usually apparent and can provide more visual information than smoke. In contrast, wildfire monitoring relies on smoke detection methods more than flame detection, since smoke usually arises earlier than flames at the beginning of fires. Approaches of flame and smoke detection are relatively independent as well as closely interrelated. Smoke can be utilized to assist the detection of flames [11] to establish robust systems. However, dense smoke sometimes hinders the monitoring of flames. In such cases, most researchers treat them as two different problems and propose diverse algorithms of computer vision and machine learning to solve them. In this thesis, only the task of flame detection is explored.

1.2 Aim and Objectives

The aim of this thesis is the accurate detection of flames in diverse environments using only colour video sequences based on computer vision and machine learning methods. The main objectives are listed below according to the aim.

- Explore and develop features which can describe the dynamic attribute of flame regions, to distinguish flames from other objects based on the temporally changing patterns.

- Establish a novel statistical model trained on various flame pixels to accurately describe the diverse colours of flames under different circumstances.

- Develop a framework to effectively distinguish flame regions from other objects based on the features extracted by deep convolutional neural network (CNN)s.

- Develop a robust framework for accurate flame detection in various environments by combing several features which describe different properties of flames. Machine learning, computer vision, and signal processing related algorithms can be used to enhance the performance of detection.

1.3 Contributions and Outline of the Thesis

The thesis is organized into six chapters. A brief overview of each chapter and the corresponding contributions are given below.

Chapter 2: This chapter overviews the existing work on vision-based flame detection methods, and briefly introduces the background knowledge of the Bayesian inference and CNNs, which are related to the proposed work in this thesis. The approaches of flame detection are summarized into two main categories: (i) the methods based on manually designed features and rules, and (ii) the deep learning based frameworks. Both groups of techniques are reviewed and discussed, with some of the widely used schemes introduced in details.

Chapter 3: A hybrid framework for flame detection is proposed in this chapter based on the combination of several features of flames. The optical flow of each pixel is estimated by the Horn-Schunck algorithm [15] to reflect its motion. Based on the estimated optical flows, a motion saliency map can be obtained using a probabilistic method of saliency analysis. It can select probable flame regions together with an intensity-based saliency map and several chromatic selective rules of flames. The temporal records of candidate flame pixels are further processed by a wavelet transform based analysis scheme. The work presented in this chapter has been published in [16]. The main contributions include:

- The probabilistic saliency analysis algorithm is employed to select salient regions whose intensity values and motion are different from surroundings, based on the intensity values and magnitudes of estimated optical flows of pixels. As such, candidate flame pixels are selected since flames are mostly dynamic and brighter than the background.

- The temporal record of each candidate pixel over several successive frames is processed by the wavelet transform based filters for its high-frequency sub-signals. Features based on the sub-signals can effectively describe the quasi-periodic behaviours of the pixels at the boundaries of flame regions, and thus reduce the false detection rate significantly.

Chapter 4: In this chapter, a flame colour model is developed based on the Dirichlet process Gaussian mixture model (DPGMM). The distribution of flame colours is modelled by a Gaussian mixture model (GMM) with the prior of its parameters set to a Dirichlet process (DP). Inference is accomplished by both Markov chain Monte Carlo (MCMC) and variational inference (VI) algorithms. Part of the work in this chapter has been published in [14].

The key contributions are listed as follows:

- The colours of flames are modelled by a GMM to handle the diversity resulted from different illumination, burning material, and intensities of combustion. The DP prior enables the model to learn all the parameters of the GMM from the training data, including the number of Gaussian components. As such, the distribution of flame colours can be estimated more accurately and efficiently compared with the conventional methods which set the number of mixture components empirically.

- The inference is implemented by two algorithms, i.e. the MCMC and VI, to manage the different quantities of available training data. The colour models trained by these two algorithms are tested on flame images of various environments, of which the results are compared and discussed.

- The trained model of flame colours is incorporated into the framework introduced in Chapter 3 for flame detection in videos. The developed DPGMM based model works effectively in distinguishing flame pixels of various colours from other pixels. The framework which includes the trained chromatic model of flames as one sub-phase is tested on various videos and achieves frame-wise accuracy higher than 95%.

Chapter 5: Two frameworks for flame detection are proposed in this chapter based on the flame region based CNN (R-CNN) and faster R-CNN, respectively. In the framework of flame R-CNN, a novel flame proposal generation scheme is developed to select probable regions of flames by utilizing the dynamic and colour properties. In the framework using faster R-CNN, the flame proposals are generated by a region proposal network. The proposals generated by either the flame proposal generation scheme or the region proposal network will be subsequently projected onto a feature map produced by several convolutional layers, and generate small feature maps of a fixed size based on a region of interest (RoI) pooling layer. The small feature maps will be further processed by additional layers to output detected regions of flames.

Novelties of this chapter are as follows:

- The flame proposal generation scheme is proposed based on the combination of the online robust principal component analysis (OR-PCA) algorithm and DPGMM based flame colour model. It considers the dynamic and colour properties of flames which are effective in generating proposals containing flames. Additionally, proposals are generated using a grid of boxes of a single aspect ratio, which eliminates the changes in the shape and texture related features caused by the RoI pooling layer and consequently enhances the performance of the framework.

- In the flame proposal generation scheme, the moving regions are effectively detected by the OR-PCA algorithm, which is robust to noise and works in an online way. The algorithm is performed on the R channel of a colour frame instead of the grayscale image (which is a common choice) to solve the difficult problem of detecting the motion of weak flames whose colours are semi-transparent. Specifically, the background behind weak flames are visible if the colours of flames are nearly transparent, which leads to only small changes in the intensity values. In contrast, the values of the R channel of flame pixels usually vary significantly with time, since the red colour is dominant in regions of flames.

- Features are extracted from probable regions of flames by convolutional layers and the RoI pooling layer. It achieves better performance than raw CNNs because it prevents the information of small-sized flame regions to be overwhelmed by that of a cluttered background. Additionally, whole frames are processed by convolutional layers, which reduces the repetitive computation caused by overlapping RoIs and thus accelerates the detection process.

- The faster R-CNN is embedded in the framework of flame detection. The influence of the diverse appearance of flames is explored on the choices of anchor boxes and performance of detection.

Chapter 6: All the methods proposed in the thesis are summarized in this chapter, together with the analyses of the corresponding results. Based on them, directions and ideas for future work are presented subsequently.

1.4 Associated Publications

The work presented in this thesis has been published in the following papers.

- **Journal Papers**

  - Zhenglin Li, Lyudmila S Mihaylova, Olga Isupova, and Lucile Rossi, “Autonomous flame detection in videos with a Dirichlet process Gaussian mixture color model,” *IEEE Trans. on Ind. Informat.*, vol. 14, no. 3, pp. 1146-1154, 2017. Impact Factor: 5.43.
  - Zhenglin Li, Lyudmila S Mihaylova, "Flame detection in videos based on flame R-CNN," to be submitted to *IEEE Trans. Inf. Forensics Security.*

- **Peer-reviewed Conference Paper**

  - Zhenglin Li, Olga Isupova, Lyudmila Mihaylova, and Lucile Rossi, “Autonomous flame detection in video based on saliency analysis and optical flow,” in *Proc. Int. Conf. Multisensor Fusion and Integration*, Baden-Baden, Germany, 2016, pp. 218-223.

Chapter 2

Literature Review

2.1 Computer Vision Methods for Flame Detection in Videos

Considerable research on flame detection has been carried out during the past decades, which contributes to the fast development of this area. Similar to other visual tasks of detection and classification, the methods of flame detection include two main groups, i.e. the **conventional frameworks based on manually designed features and rules**, and the **end-to-end approaches employing deep neural networks**.

The former group of methods usually develop various rules or extract diverse features, based on the background knowledge of flames. Fusion, either in feature-level or result-level, is commonly employed to enhance the robustness and effectiveness of the developed frameworks. The systems will provide either frame-wise decisions of the existence of flames or detect regions containing flames, according to the designed rules or features together with classifiers.

To achieve satisfactory results of detection, most existing work designs features based on the distinguishable properties of flames, such as colour [3, 4, 17, 18], texture [19, 20], and shape [21], which enable flames to be distinguished from common distracting objects, for example, lights and pedestrians in red. Additionally, all the visual attributes of flames mentioned above vary significantly with time due to the airflow caused by heat [3], so features describing the dynamic characteristic play an important role in flame detection in videos [5, 18, 19, 22]. Instead of a single feature, the combination of multiple features usually enhances the performance considerably, and thus is widely used to obtain reliable results [23].

In order to decrease the computation cost and alleviate the disturbance of non-flame objects, a motion detection phase is commonly utilized as a preprocessing step to filter out static regions, for example, the sun or steady lights. Many background subtraction approaches have been widely embedded into flame detection systems, such as adaptive background subtraction [24, 5], GMM based background subtraction [25, 26] and motion history image [27]. The first two approaches can consider slow changes of the background, and thus are relatively robust to noise and the changes of environment, while the motion history image assumes the background is static and requires images of the background, which limits its application.

Apart from the methods of motion detection, colour models of flames are widely employed in the published literature as well and have been proven effective and efficient in the selection of candidate flame pixels. The challenges of the colour models mainly come from the diversity in the colours of flames, which is resulted from the various burning material, different intensities of combustion, and the existence of smoke. Many rules or models have been proposed to describe the various colours of flames. In 2004, Chen et al. proposed a set of selective rules in the RGB colour space to describe the chromatic properties of flame pixels [3]. However, it is not robust to the changes of luminance or the influence of smoke, because of the empirical constant thresholds employed by the model. To overcome this disadvantage, Celik and Demirel developed a colour model of flames in the YCbCr space, which includes both empirical rules and polynomial models trained from data [4]. Inspired by the improvement in the performance resulted from data-driven models, an increasing number of researchers started to work on training models on real flame pixels. In [5], flame colours are described by a GMM of which the number of mixture components is set empirically. Additionally, Wang et al. proposed a flame colour model in the YCbCr colour space by modelling the ratio of the Cb and Cr values of flame pixels with a univariate Gaussian distribution [28]. Different colour models are evaluated and compared in [29, 30].

A number of candidate frame pixels or regions can be obtained after applying the background subtraction methods and/or the flame colour models. For reliable results, features which describe the dynamic property of flames are usually extracted to further verify the existence of flames in candidate regions. These features can be divided into two main categories: the ones in the temporal-space domain and those in the frequency domain.

As a conventional method extracting features in the frequency domain, Yamagishi and Yamaguchi extracted the space-time contour of flame regions in the polar coordinate system and employed the two-dimensional Fourier transform to describe its fluctuation property [18]. It works well if given concisely detected contours of flames, which are not always available in a cluttered background. Similarly, Toreyin et al. described the flickering property of flames by analysing the temporally changing patterns of flame pixels in the wavelet domain and achieved good performance [5].

Among the frameworks which extract features in the temporal-space domain, Habiboglu et al. developed features based on temporally extended covariance descriptors to detect flames [19]. Additionally, Mueller et al. proposed two novel optical flow estimation methods to describe the dynamic characteristics of combustion regions in [22].

Final decisions of the existence of flames need to be made based on the extracted features. Researchers usually employ different classifiers to reduce the false alarm rate and enhance the detection accuracy, such as the methods of SVM [31], shallow neural network [18, 22, 32], fuzzy finite automata (FFA) [33], and AdaBoost [34]. However, some internal drawbacks of the classifiers still hinder the accurate detection of flames. For example, the SVM classifier is sensitive to outliers, and the FFA method requires strong assumptions of the initial states, which limits its application to various environments. Apart from classifiers, some research estimates the probabilities of flames in the scene and makes hard decisions by setting thresholds [28, 35, 36].

Usually, multiple rules or models are combined to enhance the accuracy of detection. Averaging the probabilities of flames estimated by each sub-method is the most common way utilised by the probabilistic group of methods [28, 35, 36]. In contrast, for the frameworks which perform hard classification, the weighted voting rule has been the choice of many researchers [21]. Furthermore, Gunay et al. improved the weighted voting rule by proposing an entropy function to adjust the weights of each sub-algorithm adaptively [37]. Figure 2.1 illustrates the general framework employed by most rule and feature based methods for flame detection.

Although deep neural networks have achieved enormous success in many tasks in the area of computer vision and machine learning, they have not been widely used in the detection of flames. Among the existing research, the CNN has been employed to do the frame-wise classification of flames, in a similar way to other tasks of classification [2, 38]. Muhammad et al. [2] used an architecture of CNN named SqueezeNet [39] for flame detection, which has fewer parameters than other networks, e.g. Alexnet. Additionally, a generative adversarial network (GAN) is employed to solve the problem of training data shortage in flame detection in [38]. However, almost all the existing work using deep neural networks ignores the crucial dynamic property of flames. Besides, flame detection suffers more from the diversity of appearance than other tasks of image classification, which requires improvement based on the characteristics unique to flames.

Apart from enhancing the accuracy of detection using videos, some researchers focus on different problems. For example, the research on fire detection based on unmanned aerial vehicles pays attention to image sequence stabilisation as well [40, 41]. Additionally, fire detection can also be performed based on hyperspectral images obtained by satellites [42] instead of optical videos.

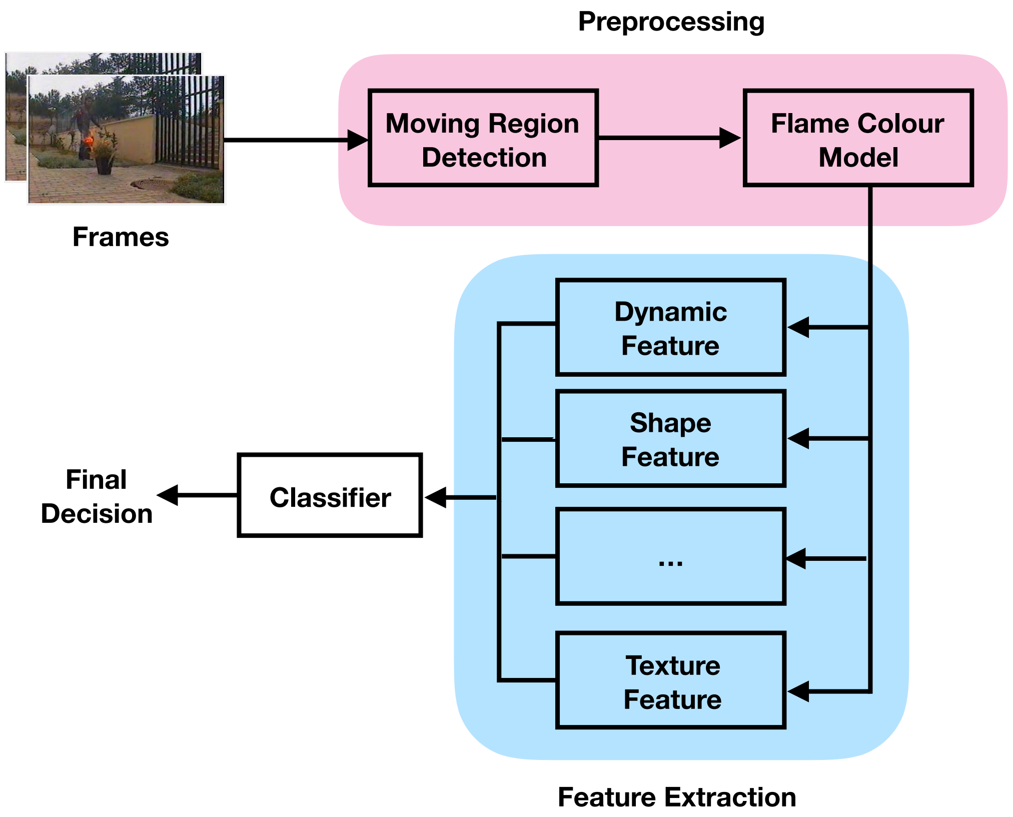


Figure 2.1. A general framework of feature and rule based methods

2.1.1 The Challenges of Flame Detection

The main challenges of the task of flame detection are listed as follows.

- The appearance of flames is rich in diversity, which can be seen from the example images shown in Figure 2.2. The diversity in colour, texture, and shape comes from various burning material, different intensities of combustion, changing environmental illumination, and the non-rigid property of flames. Therefore, the task of flame detection requires more training data and powerful features to describe the properties of flames in a high semantic level compared with the tasks of rigid object detection, such as the detection of pedestrians and vehicles.

- Flames sometimes occupy only small regions of the scene if fires happen in distant places to cameras, or fires are at the starting stage of combustion. This makes it challenging to distinguish the small regions of flames from cluttered surroundings accurately and timely. The features related to flames will be easily overwhelmed by those of more salient objects or a cluttered background. Furthermore, the various appearance also hinders locating probable regions of flames for further processing.

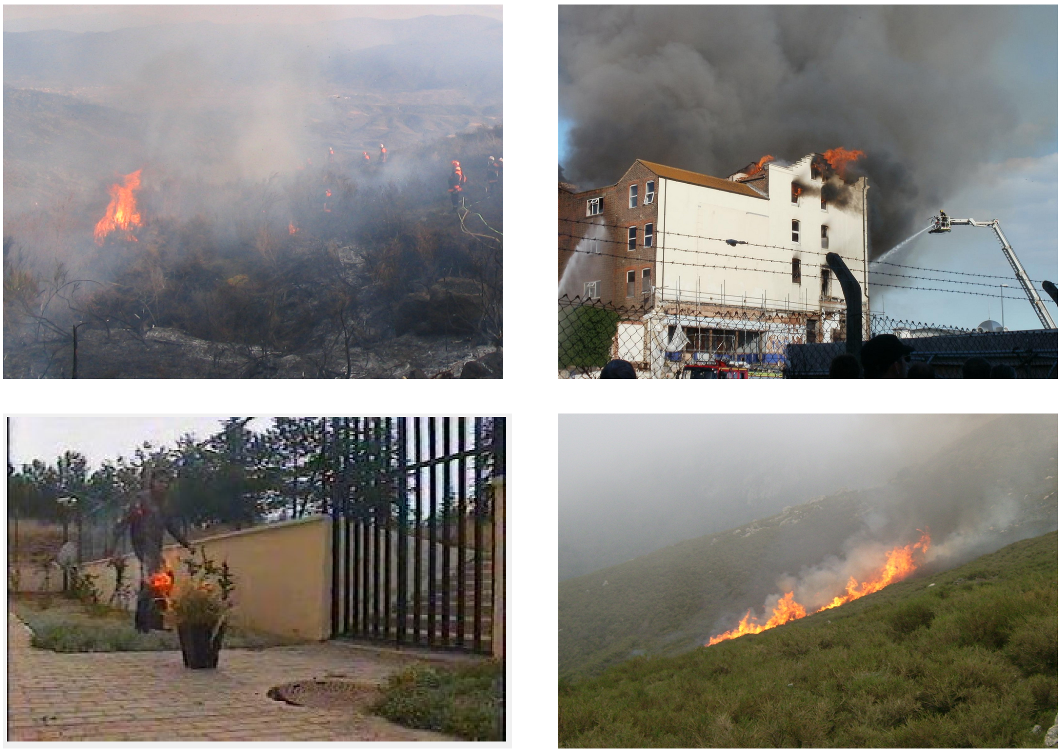


Figure 2.2. Sample images of flames of diverse appearance.

- A few flames are semi-transparent when they burn on some special fuel or at certain periods of combustion, resulting in the texture of flames being partially mixed with the background behind them [3]. Consequently, the features of flames and the background are mixed, that sometimes leads to failures in flame detection.
- Flames change with time rapidly and dramatically, making it difficult to utilize the information from adjacent frames to assist in the detection of flames in the current frame.
- Usually, a large amount of interference exists where the vision-based flame detection systems work. In this case, the approaches must keep the false alarm rate at a low level to reduce the interruption to people's work and daily life.

2.2 Background Subtraction Methods

For most tasks of video analysis, separating the moving targets from a steady background is usually the first step. It is of crucial importance because it can alleviate the influence of cluttered background and reduce the computational cost of further processing. The background subtraction methods are widely used for solving this problem [43–45].

Flame regions are dynamic due to wind or the upward air flows caused by heat while most other parts of each frame are static [3]. It will significantly reduce the computational burden if only moving regions are processed by subsequent steps of flame detection frameworks. Therefore, background subtraction is widely employed as the first phase in the methods of flame detection before feature extraction and classification. However, the representation of the background can be challenging because of the gradual changes in illumination and the existence of noise. To accurately distinguish foreground objects from the background in each frame of videos, the employed background subtraction method should be adaptive to environmental changes and robust to noise. In this section, two widely employed approaches are introduced briefly.

2.2.1 Recursive Background Subtraction Method

The recursive background subtraction algorithm works efficiently and effectively in distinguishing foreground regions from the background [46]. It achieves good performance since it is adaptive to gradual changes in the background. The approach is described as follows.

Let $V(x_s, y_s, t)$ and $B_g(x_s, y_s, t)$ denote the intensity value and the estimated background of the pixel located at (x_s, y_s) (the y_s-th row and x_s-th column) in the t-th frame, respectively. Then the background model is updated recursively as

$$B_g(x_s, y_s, t+1) = \begin{cases} \widetilde{a}\ B_g(x_s, y_s, t) + (1-\widetilde{a})V(x_s, y_s, t) & \text{if } (x_s, y_s) \text{ is stationary} \\ B_g(x_s, y_s, t) & \text{otherwise} \end{cases}, \tag{2.1}$$

where $\widetilde{a}$ is a parameter on the open interval of zero to one, determining the updating speed of the background model. Initially, $B_g(x_s, y_s, 1)$ is set to $V(x_s, y_s, 1)$. A pixel is considered as temporally static if it satisfies

$$|V(x_s, y_s, t) - V(x_s, y_s, t-1)| < T_b(x_s, y_s, t), \tag{2.2}$$

where $|\cdot|$ denotes the absolute value function, and $T_b(x_s, y_s, t)$ is the threshold for the pixel located at (x_s, y_s) in the t-th frame. It is updated as

$$T_b(x_s, y_s, t+1) = \\ \begin{cases} \widetilde{a}\ T_b(x_s, y_s, t) + (1-\widetilde{a})(c|V(x_s, y_s, t) - B_g(x_s, y_s, t)|) & \text{if } (x_s, y_s) \text{ is stationary} \\ T_b(x_s, y_s, t) & \text{otherwise} \end{cases}, \tag{2.3}$$

where c is a real number greater than 1. The threshold $T_b(x_s, y_s, t)$ updates more quickly with the changes of intensity if a greater c is set. The initial threshold T_b is set to a certain value which is the same for all pixels.

The adaptive background model ignores the small changes of intensities between frames and only takes the regions with relatively large motion as the foreground. It works effectively as the first phase to help select candidate flame regions in many frameworks of flame detection, because the motion of flames, especially violently burning ones, is much larger than that of a stationary background.

2.2.2 Adaptive Background Subtraction Based on Gaussian Mixture Models

Stauffer and Grimson [26] proposed an adaptive background subtraction method by modelling the temporal record of each pixel with a GMM and updating it using an online approximation scheme.

The model takes the temporal series of a pixel as a 'pixel process'. Specifically, for a pixel located at (x_s, y_s) of the t-th frame, its history can be expressed as

$$\{\mathbf{x}_i = [V_R(x_s, y_s, i), V_G(x_s, y_s, i), V_B(x_s, y_s, i)]^\intercal\}_{i=1}^{t} \tag{2.4}$$

where V_R, V_G and V_B denote the intensity values of the three channels of the RGB colour space, respectively.

As the intensity values of recent pixels have more information of the current background, the method models the recent history $\{\mathbf{x}_1, ..., \mathbf{x}_t\}$ of each pixel with a mixture of K Gaussian distributions. The probability of the current pixel $\mathbf{x}_t$ is

$$p(\mathbf{x}_t) = \sum_{k=1}^{K} \omega_{k,t}\ \mathcal{N}(\mathbf{x}_t; \boldsymbol{\mu}_{k,t}, \boldsymbol{\Sigma}_{k,t}), \tag{2.5}$$

where K is the number of Gaussian components (predetermined, usually set to 3 to 5), $\omega_{k,t}$, $\boldsymbol{\mu}_{k,t}$, and $\boldsymbol{\Sigma}_{k,t}$ are the estimated weight, mean and covariance matrix of the k-th Gaussian component of frame t, respectively, and $\mathcal{N}(\cdot)$ is the Gaussian probability density function defined by

$$\mathcal{N}(\mathbf{x}_t; \boldsymbol{\mu}_{k,t}, \boldsymbol{\Sigma}_{k,t}) = \frac{1}{(2\pi)^{\frac{3}{2}}\{\det(\boldsymbol{\Sigma}_{k,t})\}^{\frac{1}{2}}} \exp\{-\frac{1}{2}(\mathbf{x}_t - \boldsymbol{\mu}_{k,t})^\intercal \boldsymbol{\Sigma}_{k,t}^{-1}(\mathbf{x}_t - \boldsymbol{\mu}_{k,t})\}, \tag{2.6}$$

where $\det(\boldsymbol{\Sigma}_{k,t})$ and $\boldsymbol{\Sigma}_{k,t}^{-1}$ denote the determinant and inverse of the covariance matrix $\boldsymbol{\Sigma}_{k,t}$, respectively.

Usually, it is assumed that the three channels of the RGB colour space are independent and share the same variance to reduce the computational complexity. Therefore, the covariance matrix can be expressed as follows

$$\boldsymbol{\Sigma}_{k,t} = \sigma_{k,t}^2\ \mathbf{I}_3, \tag{2.7}$$

where $\mathbf{I}_3$ is an identity matrix of dimension 3, and $\sigma_{k,t}$ is the standard deviation of the k-th Gaussian component of the GMM of frame t. The assumption is not true in most cases, but it reduces the computational burden even though it loses some accuracy.

When a new pixel appears, it can be represented by one of the Gaussian components and be employed to update the GMM. Instead of training the model again each time a new pixel is available using the conventional expectation-maximization algorithm, the method employs an on-line K-means approach to update the model. When a new value of a pixel process appears, it will be checked whether it matches one of the K Gaussian components. The pixel is defined as matched to a component if its value is within 2.5 standard deviations to the mean. If the new pixel value does not belong to any of the existing Gaussian components, the component with the smallest probability will be replaced by a new Gaussian component. The mean of the new Gaussian component is set to the value of the current pixel with a large variance and a low weight.

The weight $\omega_{k,t}$ of the k-th distribution is updated after processing the current pixel in frame t as

$$\omega_{k,t} = (1 - \widetilde{\alpha})\omega_{k,t-1} + \widetilde{\alpha}(\widetilde{M}_{k,t}), \tag{2.8}$$

where $\widetilde{\alpha}$ is a learning parameter determining the updating speed, and $\widetilde{M}_{k,t}$ is 1 for the matched component while 0 for the others. The weights are normalized subsequently.

The means $\{\boldsymbol{\mu}_k\}$ and variances $\{\sigma_k^2\}$ for unmatched Gaussian components do not change, while the parameters of the matched one are updated as

$$\boldsymbol{\mu}_{\hat{k},t} = (1 - \rho)\boldsymbol{\mu}_{\hat{k},t-1} + \rho\mathbf{x}_t\ , \tag{2.9}$$

$$\sigma_{\hat{k},t}^2 = (1 - \rho)\sigma_{\hat{k},t-1}^2 + \rho(\mathbf{x}_t - \boldsymbol{\mu}_{\hat{k},t})^{\mathsf{T}}(\mathbf{x}_t - \boldsymbol{\mu}_{\hat{k},t}), \tag{2.10}$$

where the learning parameter ρ is defined as

$$\rho = \widetilde{\alpha}\ \mathcal{N}(\mathbf{x}_t; \boldsymbol{\mu}_{\hat{k},t-1}, \sigma_{\hat{k},t-1}^2), \tag{2.11}$$

The background is generally stationary, so the Gaussian components belonging to the background are likely to have the largest weights and smallest variance. It is because a new object will occlude the background, which will either create a new component with a low weight or increase the variance of an existing Gaussian component. Therefore, the Gaussian components are reordered according to the value of ω/σ each time after the update of the parameters, which considers both the weights and standard deviations. Then the background of the t-th frame is modelled by the first b_t^* Gaussian components which satisfy

$$b_t^* = \arg\min_b (\sum_{k=1}^{b} \omega_{k,t} > \tau_\omega), \tag{2.12}$$

where τ_ω is a predetermined threshold denoting the minimum portion of the background in a frame.

This model is updated efficiently by adopting the changes of the background. When new objects appear in the background, the model does not need to be completely reconstructed.

2.3 Colour Models in Flame detection

Colour is among the most important characteristics of flames. Colour models work effectively in selecting candidate flame pixels as an intermediate step. However, the colours of flames are of wide diversity, which makes it challenging to establish a model or a set of selective criteria. In this section, two state-of-the-art chromatic selective rules of flames are introduced.

2.3.1 Chromatic Selective Rules in the RGB Colour Space

Chen et al. [3] proposed three rules in the RGB colour space to describe the colour property of flames. According to their discussion in [3], the colours of flames generally belong to the red-yellow range, and the hue of flame-coloured pixels is mapped to the range of 0° to 60°. Therefore, the RGB values of flames are considered to satisfy the condition given by:

$$V_R \geqslant V_G > V_B, \tag{2.13}$$

where V_R, V_G and V_B denote the intensity values of the three channels of the RGB colour space, respectively.

The colour of red is dominant in the regions of flames, so it is reasonable to set a threshold τ_R for the R values. Pixels of which the R values are greater than the threshold are likely to be part of flames. Therefore, the second rule is

$$V_R > \tau_R. \tag{2.14}$$

To relieve the influence of illumination, another rule is set as follows

$$V_S > (255 - V_R) * \tau_S/\tau_R \tag{2.15}$$

where V_S denotes the value of the saturation channel of the HSV colour space, and τ_S is a predetermined threshold of V_S.

The selective rules can detect most flame-coloured pixels if the pre-determined thresholds are set appropriately. However, they are not robust to the changes in environmental illumination because of the constant thresholds and the inherent disadvantage of the RGB colour space. Specifically, the three channels of the RGB space are not independent of brightness, and thus are susceptible to the changes of illumination.

2.3.2 Chromatic Selective Rules in the YCbCr Colour Space

To overcome the disadvantages of the chromatic rules introduced in Section 2.3.1, Celik and Demirel [4] proposed a group of rules in the YCbCr colour space. There are three channels in the YCbCr colour space, i.e. the luminance Y, the blue-difference chroma Cb and the red-difference chroma Cr, respectively. For a given image, the mean value of each channel is defined as follows

$$\bar{V}_Y = \frac{1}{N}\sum_{x_s,y_s} V_Y(x_s,y_s), \tag{2.16}$$

$$\bar{V}_{Cb} = \frac{1}{N}\sum_{x_s,y_s} V_{Cb}(x_s,y_s), \tag{2.17}$$

$$\bar{V}_{Cr} = \frac{1}{N}\sum_{x_s,y_s} V_{Cr}(x_s,y_s), \tag{2.18}$$

where N denotes the number of pixels in a frame, and V_Y, V_{Cb} and V_{Cr} represent the values of the Y, Cb and Cr channels, respectively.

Given a pixel located at an arbitrary location, e.g. (x_s, y_s), it is likely to be flame-coloured if the following rules are satisfied.

$$V_Y(x_s,y_s) > V_{Cb}(x_s,y_s), \tag{2.19}$$

$$V_{Cr}(x_s,y_s) > V_{Cb}(x_s,y_s), \tag{2.20}$$

$$V_Y(x_s, y_s) > \bar{V}_Y, \tag{2.21}$$
$$V_{Cb}(x_s, y_s) < \bar{V}_{Cb}, \tag{2.22}$$
$$V_{Cr}(x_s, y_s) > \bar{V}_{Cr}, \tag{2.23}$$

$$|V_{Cb}(x_s, y_s) - V_{Cr}(x_s, y_s)| \geqslant \tau, \tag{2.24}$$

where τ is an empirical threshold, and is set to 0.4 according to the ROC curve proposed in [4].

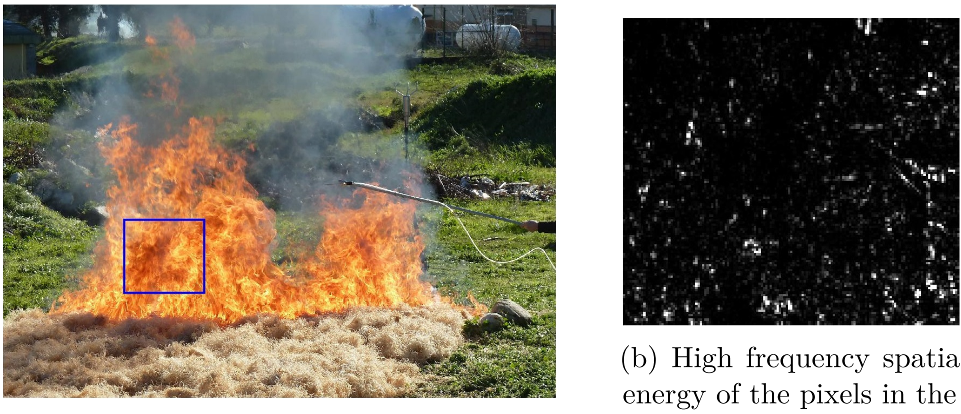
Besides the above rules, the method performs a statistical analysis of the colours of the flame pixels which are manually annotated over a database. They employed the intersections of three polynomials to model the area of flame pixels in the plane of Cb-Cr. The three polynomials, which are denoted by $f_u(V_{Cr})$, $f_l(V_{Cr})$, and $f_d(V_{Cr})$, respectively, are derived with the least squares estimation algorithm. The estimated polynomials are as follows

$$\begin{aligned} f_u(V_{Cr}) =& -2.6 \times 10^{-10} V_{Cr}^7 + 3.3 \times 10^{-7} V_{Cr}^6 - 1.7 \times 10^{-4} V_{Cr}^5 + 5.16 \times 10^{-2} V_{Cr}^4 \\ & - 9.10 \times V_{Cr}^3 + 9.60 \times 10^2 V_{Cr}^2 - 5.60 \times 10^4 V_{Cr} + 1.40 \times 10^6, \quad (2.25) \\ f_l(V_{Cr}) =& -6.77 \times 10^{-8} V_{Cr}^5 + 5.50 \times 10^{-5} V_{Cr}^4 - 1.76 \times 10^{-2} V_{Cr}^3 + 2.78 V_{Cr}^2 \\ & - 2.15 \times 10^2 V_{Cr} + 6.62 \times 10^3, \quad (2.26) \\ f_d(V_{Cr}) =& 1.81 \times 10^{-4} V_{Cr}^4 - 1.02 \times 10^{-1} V_{Cr}^3 + 2.17 \times 10 V_{Cr}^2 - 2.05 \times 10^3 V_{Cr} \\ & + 7.29 \times 10^4. \quad (2.27) \end{aligned}$$

The rules derived from the three polynomials are given by

$$V_{Cb}(x_s, y_s) \geqslant f_u(V_{Cr}(x_s, y_s)), \tag{2.28}$$
$$V_{Cb}(x_s, y_s) \leqslant f_d(V_{Cr}(x_s, y_s)), \tag{2.29}$$
$$V_{Cb}(x_s, y_s) \leqslant f_l(V_{Cr}(x_s, y_s)). \tag{2.30}$$

Pixels satisfying the rules in (2.19) - (2.30) are classified as flame-coloured pixels. This model is established in the YCbCr colour space, which separates the component of luminance from the components of chroma, and thus increases the robustness of the model to the changes of environmental illumination.



(a) Flame regions

(b) High frequency spatial wavelet energy of the pixels in the bounded region in Figure 2.3a

Figure 2.3. Flames and spatial high-frequency wavelet energy of the corresponding bounded region.

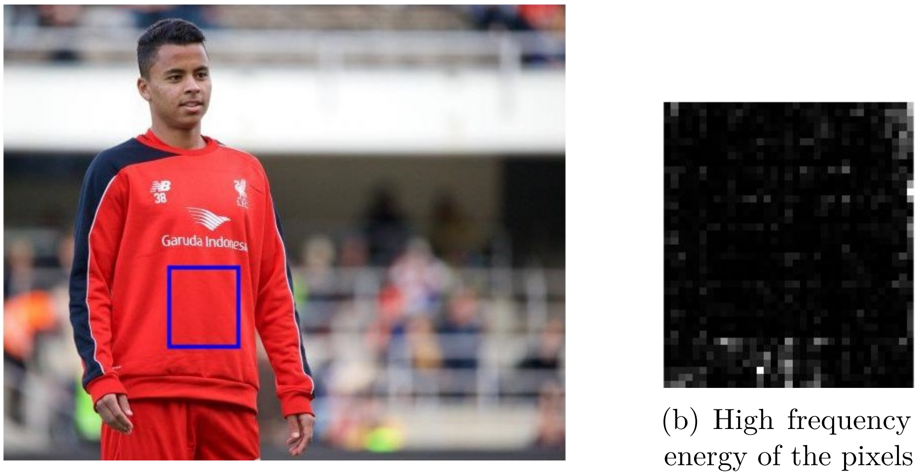
2.4 Features Describing the Dynamic and Texture Properties of Flames

2.4.1 Spatial Wavelet Transform Based Features of Flames

Toreyin et al. [5] proposed both temporal and spatial wavelet transform based features to describe the dynamic and texture characteristics of flames, respectively, which achieve good performance in the detection of flames. The temporal analysis based on the wavelet transform will be introduced in details in Section 3.4. In addition to the temporal analysis, a two-dimensional discrete wavelet transform is performed to extract texture-related features that can be used to distinguish flames from non-flame objects.

The high-high, high-low, low-high and low-low sub-images are generated by the 2-D wavelet transform. All the sub-images are of quarter-size of the original frame, so they are up-sampled in both horizontal and vertical directions by a scaling factor of 2, to obtain images with the same size as the original frame. The up-sampled high-high, low-high and high-low sub-images in the wavelet domain are denoted by $\mathbf{S}_{hh}$, $\mathbf{S}_{lh}$ and $\mathbf{S}_{hl}$, respectively. Given a pixel located at (x_s, y_s), the high-frequency spatial wavelet energy is defined as $[\mathbf{S}_{hh}(x_s, y_s)]^2 + [\mathbf{S}_{lh}(x_s, y_s)]^2 + [\mathbf{S}_{hl}(x_s, y_s)]^2$. Obvious differences can be observed from Figure 2.3 and Figure 2.4, in which the high-frequency spatial wavelet energy of the pixels of flames and flame-coloured objects are shown. It shows that the spatial variations in regions of flames are more significant than those in the regions of flame-coloured objects.

To reduce the computational complexity, a feature is designed for candidate flame regions instead of pixels. Specifically, the feature is defined as the average



(a) Flame-coloured object

(b) High frequency spatial wavelet energy of the pixels in the bounded region in Figure 2.4a

Figure 2.4. Flame-coloured object and spatial high-frequency wavelet energy of the corresponding bounded region.

high-frequency spatial wavelet energy in a candidate region of flames Ω_c as follows

$$\hat{W} = \frac{1}{N} \sum_{(x_s,y_s)\in\Omega_c} \left\{ [\mathbf{S}_{hh}(x_s,y_s)]^2 + [\mathbf{S}_{lh}(x_s,y_s)]^2 + [\mathbf{S}_{hl}(x_s,y_s)]^2 \right\}, \tag{2.31}$$

where N is the number of pixels in the candidate flame region Ω_c.

2.4.2 Covariance Descriptor Based Features of Flames

Habiboglu et al. [19] proposed a temporally extended covariance descriptor to describe the dynamic and texture properties of flames. The method divides frames into rectangular regions of fixed sizes and extracts temporally extended covariance descriptor based features from each region for classification.

Given a pixel located at (x_c, y_c) at time t_c. A property vector $\boldsymbol{\eta}(x_c, y_c, t_c)$ of the pixel is defined by

$$\boldsymbol{\eta}(x_c, y_c, t_c) = \begin{bmatrix} V_R(x_c, y_c, t_c) \\ V_G(x_c, y_c, t_c) \\ V_B(x_c, y_c, t_c) \\ V(x_c, y_c, t_c) \\ \dot{V}_x = \left|\frac{\partial V(x_c, y_c, t_c)}{\partial x_c}\right| \\ \dot{V}_y = \left|\frac{\partial V(x_c, y_c, t_c)}{\partial y_c}\right| \\ \ddot{V}_{xx} = \left|\frac{\partial^2 V(x_c, y_c, t_c)}{\partial x_c^2}\right| \\ \ddot{V}_{yy} = \left|\frac{\partial^2 V(x_c, y_c, t_c)}{\partial y_c^2}\right| \\ \dot{V}_t = \left|\frac{\partial V(x_c, y_c, t_c)}{\partial t_c}\right| \\ \ddot{V}_{tt} = \left|\frac{\partial^2 V(x_c, y_c, t_c)}{\partial t_c^2}\right| \end{bmatrix}, \tag{2.32}$$

where $V_R(x_c, y_c, t_c), V_G(x_c, y_c, t_c)$ and $V_B(x_c, y_c, t_c)$ denote the values of the R, G and B channels of the pixel located at (x_c, y_c) at time t_c, respectively. Furthermore, V denotes the intensity, and $\dot{V}_x, \dot{V}_y$ and $\dot{V}_t$ represent the first order derivatives of V with respect to x_c, y_c and t_c, respectively. Similarly, $\ddot{V}_{xx}, \ddot{V}_{yy}$ and $\ddot{V}_{tt}$ denote the second order derivatives of V with respect to x_c, y_c and t_c, respectively.

It is noteworthy that x_c, y_c, t_c and V are continuous variables. However, the coordinates and intensity values of pixels are discrete in frames of digital videos. Therefore, the first and second order partial derivatives of pixels are computed by filtering the image with the filters $[-1, 0, 1]$ and $[1, -2, 1]$, respectively.

Denote the discrete coordinates of a pixel as (x_s, y_s, t) (meaning a pixel located at the y_s-th row, x_s-th column, and the t-th frame). The covariance descriptor of a candidate flame region Ω_c is estimated as follows

$$\hat{\boldsymbol{\Sigma}}_t = \frac{1}{N} \sum_{(x_s, y_s) \in \Omega_c} (\boldsymbol{\eta}(x_s, y_s, t) - \overline{\boldsymbol{\eta}}_t)(\boldsymbol{\eta}(x_s, y_s, t) - \overline{\boldsymbol{\eta}}_t)^{\mathsf{T}}, \tag{2.33}$$

where $\overline{\boldsymbol{\eta}}_t$ is defined as

$$\overline{\boldsymbol{\eta}}_t = \frac{1}{N} \sum_{(x_s, y_s) \in \Omega_c} \boldsymbol{\eta}(x_s, y_s, t), \tag{2.34}$$

where N is the number of pixels in the candidate regions, and $\boldsymbol{\eta}(x_s, y_s, t)$ is the property vector defined in Eq. (2.32).

The feature for classification is formed by the lower or upper triangular parts of the covariance descriptor $\hat{\boldsymbol{\Sigma}}_t$ and can be classified by an SVM classifier.

2.4.3 Optical Flow Based Features of Flames

Mueller et al. [22] designed two novel algorithms for estimating the optical flows of flames, i.e., the optimal mass transport (OMT) and non-smooth data (NSD) optical flows, to extract features for flame detection based on the dynamic property.

2.4.3.1 Optimal Mass Transport Optical Flow

Different from classic optical flow models, the OMT based flow estimation is posed as a generalized mass transport problem, with the conservation rule of

$$\dot{V}_t + \nabla \cdot (\mathbf{u}V) = 0, \tag{2.35}$$

where $\mathbf{u} = (u, v)^{\mathsf{T}}$ is an optical flow vector, V denotes the intensity value of a pixel, $\dot{V}_t$ is the first-order partial derivative of V with respect to time t_c, and $\nabla \cdot (\cdot)$ is the divergence operation of a vector.

The OMT optical flow is estimated by minimizing the total energy given by

$$\min_{\mathbf{u}} \frac{1}{2} \int_{\Omega} \int_{0}^{T} (\dot{V}_t + \nabla \cdot (V\mathbf{u}))^2 + \alpha \|\mathbf{u}\|_2^2 V \; dt_c \; dx_c \; dy_c, \tag{2.36}$$

where $\|\cdot\|_2^2$ denotes the squared ℓ_2-norm, and $(x_c, y_c) \in \Omega$ are spatial coordinates along x-axis and y-axis, respectively. It is noteworthy that t_c, x_c, and y_c are continuous variables. The solution of (2.36) is

$$\mathbf{u} = (\alpha\widehat{\mathbf{V}} + \mathbf{A}^{\mathsf{T}}\mathbf{A})^{-1}(\mathbf{A}^{\mathsf{T}}\hat{b}), \tag{2.37}$$

where $\widehat{\mathbf{V}}$ denotes a matrix with the average intensity value $(V(t) + V(t-1))/2$ on its diagonal and

$$\mathbf{A} = [D_x V, \;\; D_y V], \tag{2.38}$$

$$\hat{b} = -\dot{V}_t, \tag{2.39}$$

where D_x and D_y are the central-difference sparse-matrix derivative operators.

2.4.3.2 Non-smooth Data Optical Flow

The flame regions are stable and tend to be saturated sometimes, which makes the dynamic property of flames less obvious. Therefore, another method of optical flow estimation named NSD is proposed in [22], which is designed for flames of this type. The total energy function is given by

$$\min_{\mathbf{u}} \frac{1}{2} \int_{\Omega} \int_{0}^{T} (\dot{V}_t + \nabla V \cdot \mathbf{u})^2 + \alpha \|\mathbf{u}\|_2^2 \; dt_c \; dx_c \; dy_c, \tag{2.40}$$

where ∇V denotes the gradient of intensity V.

The NSD optical flow is designed to be non-smooth because the motion of saturated flame regions is usually non-smooth. Therefore, the estimation algorithm of NSD is suitable for the detection of saturated flames.

The solutions of (2.40) are obtained as

$$u = -\frac{\dot{V}_x \dot{V}_t}{\|\nabla V\|_2^2 + \alpha} \, , \tag{2.41}$$

$$v = -\frac{\dot{V}_y \dot{V}_t}{\|\nabla V\|_2^2 + \alpha} \, , \tag{2.42}$$

where $\dot{V}_x, \dot{V}_y$ and $\dot{V}_t$ denote the first-order partial derivatives of intensity V with respect to x_c, y_c and t_c, respectively, and α is a regularization parameter.

2.4.3.3 Optical Flow Features

The two novel optical flows, i.e., OMT and NSD, do not work as features directly. Instead, a region-based feature vector is designed based on the estimated optical flows. Given a candidate flame region Ω_c, a four-dimensional feature $\mathbf{f} = (f_1, f_2, f_3, f_4)^\mathsf{T}$ is designed based on the estimated optical flows. Specifically, f_1 and f_2 are the mean magnitudes of the estimated OMT and NSD optical flows, respectively. It is believed that the values of f_1 and f_2 will be high of regions which are part of moving objects (including flames). To further distinguish flames and non-flame moving objects, two more features f_3 and f_4 are defined based on the estimated optical flows. Specifically, f_3 is designed to measure how well the OMT optical flow matches the template of an ideal source. Apart from it, the variance of the directions of NSD optical flows is defined as f_4 since flame pixels usually move in various directions, which is different from non-flame moving objects. For accurate detection of flames, a neural network is trained based on the feature vectors $\mathbf{f} = (f_1, f_2, f_3, f_4)^\mathsf{T}$.

2.5 Bayesian Inference

Bayesian inference is the process of applying Bayes' theorem to update a probability model with observed data $\mathbf{X} = \{\mathbf{x}_i\}$. Compared with the observations themselves, some latent variables $\mathbf{w}$ that specify the model, are usually of more interest. A prior distribution $p(\mathbf{w})$ is specified based on all the information of $\mathbf{w} = \{w_k\}$ which is known before obtaining any observation. Given the likelihood $p(\mathbf{x}|\mathbf{w})$, the posterior distribution of $p(\mathbf{w}|\mathbf{X})$ is estimated based on the Bayes' theorem as follows

$$p(\mathbf{w}|\mathbf{X}) = \frac{p(\mathbf{X}|\mathbf{w})p(\mathbf{w})}{p(\mathbf{X})} \tag{2.43}$$

$$= \frac{p(\mathbf{X}|\mathbf{w})p(\mathbf{w})}{\int p(\mathbf{X}|\mathbf{w})p(\mathbf{w})d\mathbf{w}}. \tag{2.44}$$

The challenge of the computation of posterior mainly lies in the integral of the joint distribution in the denominator, which is mostly intractable. Therefore, approximating the posterior distribution becomes a crucial problem in modern statistics. The MCMC and VI algorithms are two of the most widely-used methods for approximating posteriors in Bayesian statistics for the past decades. Brief descriptions of the two groups of algorithms are provided below.

2.5.1 Markov Chain Monte Carlo

Even though the posterior is difficult to calculate analytically, the quantities of interest can be approximated based on the samples drawn from it. This is the basic idea of the Monte Carlo approximation [47]. Therefore, sampling from the posterior distribution becomes a crucial problem.

Fortunately, a group of algorithms named MCMC provide a feasible way to get samples from a distribution without knowing its closed form. By constructing a Markov chain in the state space with a stationary distribution which is the same as the target distribution (posterior in this section), the quantities of interest can be approximated using the samples drawn from the Markov chain according to the Monte Carlo method. Among the MCMC methods, the algorithms of Metropolis-Hastings [47] and Gibbs sampling (GS) will be introduced below.

2.5.1.1 Metropolis Hastings Algorithm

The Metropolis-Hastings algorithm aims at sampling from the desired distribution $p^*(\mathbf{w})$, which is the posterior $p(\mathbf{w}|\mathbf{X})$ in this section. It achieves this by setting up a Markov chain which convergences to the desired distribution. The distribution should be invariant with respect to the Markov chain [48], which will be proven after the introduction of the algorithm.

Given the current state $\mathbf{w}^s$, a random candidate state $\mathbf{w}'$ can be generated according to a proposal distribution $q(\mathbf{w}'|\mathbf{w}^s)$, which describes the transition probability between states. Subsequently, a decision needs to be made on whether to accept it or not according to an acceptance probability r'. Given a symmetric proposal distribution $q(\cdot|\cdot)$ satisfying $q(\mathbf{w}'|\mathbf{w}^s) = q(\mathbf{w}^s|\mathbf{w}')$, the acceptance probability r' is defined as

$$r' = \min\left(1, \frac{p^*(\mathbf{w}')}{p^*(\mathbf{w}^s)}\right). \tag{2.45}$$

It means that $\mathbf{w}'$ is accepted with probability 1 if $p^*(\mathbf{w}') > p^*(\mathbf{w}^s)$. On the contrary, when the candidate state is less probable than the current state, i.e. $p^*(\mathbf{w}') < p^*(\mathbf{w}^s)$, there is still a chance of moving to $\mathbf{w}'$ from $\mathbf{w}^s$. The probability of accepting a less probable state is $p^*(\mathbf{w}')/p^*(\mathbf{w}^s)$. If $\mathbf{w}'$ is rejected, the new state $\mathbf{w}^{s+1}$ will be set the same as the current one, i.e. $\mathbf{w}^{s+1} = \mathbf{w}^s$.

However, asymmetric proposal distributions are chosen sometimes, meaning that $q(\mathbf{w}'|\mathbf{w}^s) \neq q(\mathbf{w}^s|\mathbf{w}')$. A Hastings correction is needed to compensate for the bias caused by the asymmetric proposal distribution, so the acceptance probability is rewritten as:

$$\begin{aligned} r' &= \min\left(1, \frac{p^*(\mathbf{w}')q(\mathbf{w}^s|\mathbf{w}')}{p^*(\mathbf{w}^s)q(\mathbf{w}'|\mathbf{w}^s)}\right) && (2.46)\\ &= \min\left(1, \frac{p^*(\mathbf{w}')/q(\mathbf{w}'|\mathbf{w}^s)}{p^*(\mathbf{w}^s)/q(\mathbf{w}^s|\mathbf{w}')}\right). && (2.47) \end{aligned}$$

As mentioned above, the distribution should be invariant with respect with the designed Markov chain, of which a sufficient (not necessary) condition is that the transition probabilities of the Markov chain satisfy the property of detailed balance defined by

$$p^*(\hat{\mathbf{w}})p(\hat{\mathbf{w}}'|\hat{\mathbf{w}}) = p^*(\hat{\mathbf{w}}')p(\hat{\mathbf{w}}|\hat{\mathbf{w}}'), \tag{2.48}$$

where $\hat{\mathbf{w}}$ and $\hat{\mathbf{w}}'$ are two states of a Markov chain. With respect to the designed Markov chain, it has

$$\begin{aligned} p^*(\mathbf{w}^s)q(\mathbf{w}'|\mathbf{w}^s)r' &= p^*(\mathbf{w}^s)q(\mathbf{w}'|\mathbf{w}^s)\min\left(1, \frac{p^*(\mathbf{w}')q(\mathbf{w}^s|\mathbf{w}')}{p^*(\mathbf{w}^s)q(\mathbf{w}'|\mathbf{w}^s)}\right) && (2.49)\\ &= \min\left(p^*(\mathbf{w}^s)q(\mathbf{w}'|\mathbf{w}^s), p^*(\mathbf{w}')q(\mathbf{w}^s|\mathbf{w}')\right) && (2.50)\\ &= p^*(\mathbf{w}')q(\mathbf{w}^s|\mathbf{w}')\min\left(1, \frac{p^*(\mathbf{w}^s)q(\mathbf{w}'|\mathbf{w}^s)}{p^*(\mathbf{w}')q(\mathbf{w}^s|\mathbf{w}')}\right), && (2.51) \end{aligned}$$

which satisfies the property of detailed balance. Therefore, the designed Markov chain converges to the desired distribution $p^*(\mathbf{w})$.

One of the most important advantages of the Metropolis-Hastings algorithm is that the exact density of $p^*(\mathbf{w})$ does not have to be known for calculating r'. Probability values up to a constant are enough as the normalisation constants are cancelled in the fraction, which solves the difficult computation problem in Eq. (2.43).

2.5.1.2 Gibbs Sampling

The widely-used GS algorithm is introduced in this section. The Gibbs scheme samples each variable iteratively conditioned on the most recent values of other variables [49]. Specifically, a new state $\mathbf{w}^{s+1}$, as a joint sample of several variables, is generated by iteratively updating the value of each component given a current sample $\mathbf{w}^{s}$.

The GS algorithm works as follows if taking a sample of dimension 3 as an example.

$$\begin{aligned} w_1^{s+1} &\sim p^*(w_1|w_2^s, w_3^s), \\ w_2^{s+1} &\sim p^*(w_2|w_1^{s+1}, w_3^s), \\ w_3^{s+1} &\sim p^*(w_3|w_1^{s+1}, w_2^{s+1}). \end{aligned}$$

The GS algorithm can be seen as a special case of the Metropolis-Hastings algorithm with an acceptance probability of 1. Instead of the standard GS method, the collapsed Gibbs sampler is employed in the training of the flame colour model. The collapsed Gibbs sampler integrates out some of the latent variables analytically, and only samples the rest variables. It will accelerate the process (usually significantly) since it reduces the dimension of the variables.

2.5.2 Variational Inference

Different from the MCMC algorithms, the VI schemes choose an optimal distribution to approximate the posterior from a family of variational distributions $\mathcal{Q}$ over latent variables $\mathbf{w}$ [50]. The distribution that is the most similar to the exact posterior is chosen as the optimal distribution $q^*(\mathbf{w})$, of which the similarity is measured by the KL divergence [47, 51]. In a nutshell, $q^*(\mathbf{w})$ is the optimal distribution that minimizes the KL divergence with respect to $p(\mathbf{w}|\mathbf{X})$ [52], i.e.,

$$q^*(\mathbf{w}) = \underset{q(\mathbf{w})\in\mathcal{Q}}{\operatorname{argmin}} \ \mathrm{KL}(q(\mathbf{w})||p(\mathbf{w}|\mathbf{X})). \tag{2.52}$$

As such, the inference becomes an optimization problem, which can be solved neatly and efficiently by many existing algorithms. The complexity of the VI scheme depends on the complexity of the variational distributions.

According to the definition of KL divergence, the objective function is given by

$$\begin{aligned} \mathrm{KL}(q(\mathbf{w} \parallel p(\mathbf{w}|\mathbf{X})) &= \mathbb{E}[\log q(\mathbf{w})] - \mathbb{E}[\log p(\mathbf{w}|\mathbf{X})] && (2.53) \\ &= \mathbb{E}[\log q(\mathbf{w})] - \mathbb{E}[\log p(\mathbf{w}, \mathbf{X})] + \log p(\mathbf{X}), && (2.54) \end{aligned}$$

where $\mathbb{E}(\cdot)$ denotes the expectation of distributions, and all the expectations are taken with respect to $q(\mathbf{w})$ in this section.

Eq. (2.54) reveals that the objective function is not computable because it relates to the evidence $p(\mathbf{X})$ which is difficult to compute. That is the reason why the posterior in Eq. (2.43) cannot be calculated directly and researchers seek for approximating methods.

Fortunately, the optimal approximate distribution is still available by optimizing an alternative objective that is equivalent to the KL divergence defined in Eq. (2.53) up to a constant. As the evidence $\log p(\mathbf{X})$ is a constant with respect to $q(\mathbf{w})$, only the other two items in Eq. (2.54) need to be considered to optimize the KL divergence. A function named ELBO is defined as follows [50]

$$\mathrm{ELBO}(q) = \mathbb{E}[\log p(\mathbf{w}, \mathbf{X})] - \mathbb{E}[\log q(\mathbf{w})]. \tag{2.55}$$

The minimization of KL divergence can be achieved by maximizing the $\mathrm{ELBO}(q)$ with respect to the variational parameters of $q(\mathbf{w})$. The ELBO can be rewritten as

$$\begin{aligned} \mathrm{ELBO}(q) &= \mathbb{E}[\log p(\mathbf{w})] + \mathbb{E}[\log p(\mathbf{X}|\mathbf{w})] - \mathbb{E}[\log q(\mathbf{w})] & (2.56) \\ &= \mathbb{E}[\log p(\mathbf{X}|\mathbf{w})] - \mathrm{KL}(q(\mathbf{w})||p(\mathbf{w})). & (2.57) \end{aligned}$$

From Eq. (2.57), it can be seen that maximizing the ELBO function will encourage the variational distribution that explains the observations well and similar to the prior.

Additionally, some algorithms, such as [53], also minimize the free energy defined by

$$\mathcal{F} = \mathbb{E}[\log q(\mathbf{w})] - \mathbb{E}[\log p(\mathbf{w}, \mathbf{X})], \tag{2.58}$$

to minimize the KL divergence, where $\mathcal{F}$ equals to the opposite of ELBO.

2.5.2.1 Mean-field Variational Family

In the family of mean-field variational distributions, each of the latent variables is independent of others and dominated by a distinct parameter (or a distinct set of parameters). A general format of mean-field variational distributions is

$$q(\mathbf{w}) = \prod_{k=1}^{K_l} q_k(w_k), \tag{2.59}$$

where K_l is the number of latent variables, and $q_k(\cdot)$ is the distinct distribution of the latent variable w_k.

2.5.3 Comparison of Markov Chain Monte Carlo and Variational Inference Algorithms

Both the MCMC and VI algorithms have been widely utilized for the inference of Bayesian statistics. Although they aim at solving the same problem, the two algorithms are suitable for different situations. Therefore, the advantages and disadvantages of these algorithms are analysed and compared in this section.

The **VI** algorithms have several **advantages** [47, 48]:

- They work more efficiently than the algorithms of MCMC, especially when applied to problems with a large quantity of data;
- It clear to decide when to stop the optimization process;
- They are deterministic algorithms.

They also suffer from some **disadvantages**:

- The derivation of VI is usually complicated;
- They utilize a distribution to approximate the target one (posterior in this section), rather than provide an accurate estimation of the distribution of interest.

In contrast, the **MCMC** algorithms have different pros and cons.

Advantages:

- They are easy to understand and implement;
- They provide samples from the exact distributions of interest;
- They can be applied to a broader range of models compared with the VI algorithms.

Disadvantages:

- They are usually computationally intensive;
- It is difficult to determine the convergence of the algorithms.

Generally, the VI algorithms are more suitable for large datasets, while the MCMC methods are more popular in situations where accurate results are preferred with small quantities of data.

2.6 Convolutional Neural Networks

2.6.1 Basic Concepts of CNN

CNNs have achieved enormous success in the area of machine learning in the past decades [54], especially in processing images [55]. They are a specialized type of neural networks which employ the operation of convolution.

A regular neural network takes a vector as input and passes it to several hidden layers, which are composed of a number of neurons [56]. Each neuron of a layer is connected to all the neurons of its previous layer in a directed way, and none of them shares any connection. It can be seen from the diagram of a regular neural network shown in Figure 2.5. Consequently, the number of the parameters of a regular neural network increases dramatically with the size of the network, leading to heavy computational and storage burdens as well as overfitting problems.

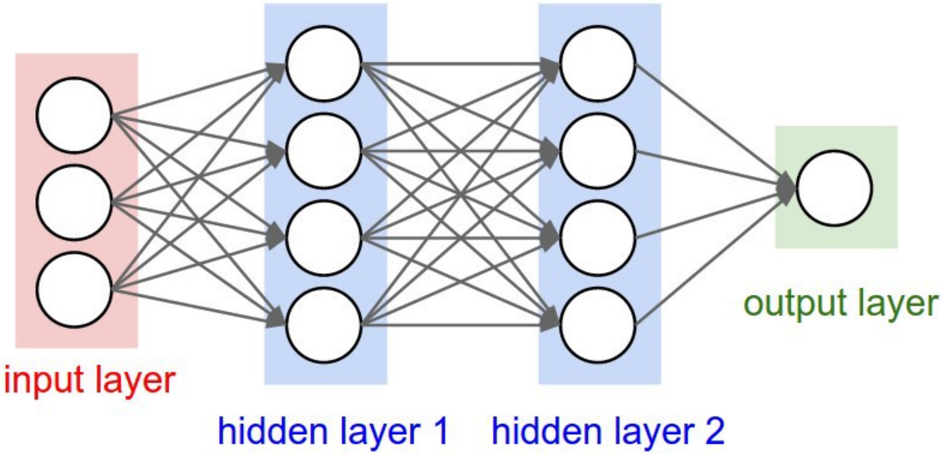


Figure 2.5. Diagram of a regular neural network [1].

The regular neural networks with full connections do not achieve satisfactory performance on visual tasks, such as image classification or object detection. One important reason is that images are usually of relatively large sizes, that leads to a huge number of parameters to be learned. For example, the size of images for most computer vision tasks is 224×224 or larger, meaning an input of size $150,528$ for a regular RGB image. Given a hidden layer of $1,024$ neurons, the network has $150,528 \times 1,024 = 154,140,672$ parameters to estimate only for the first hidden layer, which significantly hinders its applications.

Additionally, it is the local features that are crucial in most tasks of computer vision. In other words, connectivities to nearby pixels are more important than those to distant pixels in an image, meaning that fully connecting all neurons in every layer results in a big waste of computational and storage resources. Additionally, a slight translation of the targets in images will induce significantly different activations in a regular neural network, which does not fit most visual tasks, such as the classification of images.

To overcome the drawbacks of regular neural networks applied to visual tasks, CNNs adopt three vital ideas: **sparse interactions**, **parameter sharing** and **equivariant representations** [56]. The sparse interaction is implemented by setting the kernels smaller than the image size, to detect local, low-level but meaningful features. Parameter sharing enables each kernel to be employed in multiple positions of the input. It is reasonable since the low-level features, such as edges and corners, are usually shared across whole images. These two ideas significantly reduce the computational operations and storage requirements. The parameter sharing idea of CNNs also contributes to the characteristic of equivariance to translation. For example, if a target object is shifted in an input image, its representation will shift accordingly in the output.

A CNN is a sequence of specified layers, in which the neurons are arranged in 3 dimensions. The diagram of a CNN is shown in Figure 2.6. A deep CNN is mainly constructed by three types of layers: the **convolutional layer**, **pooling layer** and **fully-connected layer** [55].

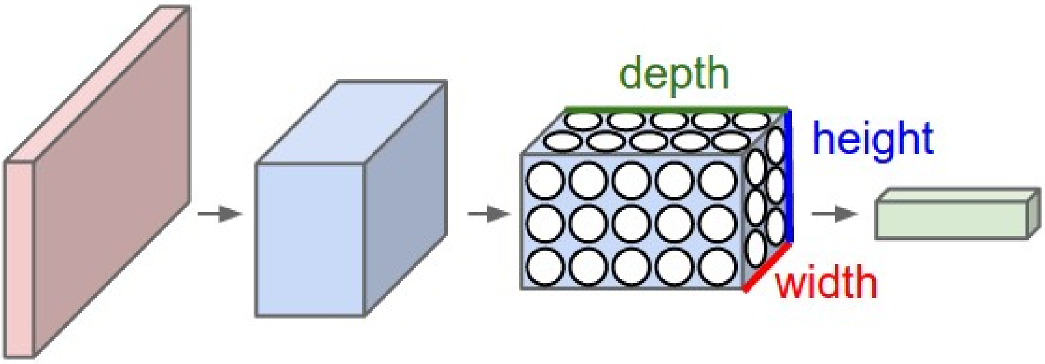


Figure 2.6. Diagram of a convolutional neural network [1].

- **Convolutional layer**: It is utilised to extract features from its input. In each convolutional layer, various kernels are employed to conduct the operation of convolution on the input images and intermediate feature maps.
- **Pooling layer**: It provides slightly modified results of the output of its previous layer in a rectangular neighbourhood of each node. It usually results in downsampling along the spatial dimensions. Widely used pooling functions include max pooling, average pooling, and L^2 norm pooling [57].
- **Fully-connected layer**: Units in this layer are fully connected to all activations of its previous layer, which is the same as the connections in regular neural

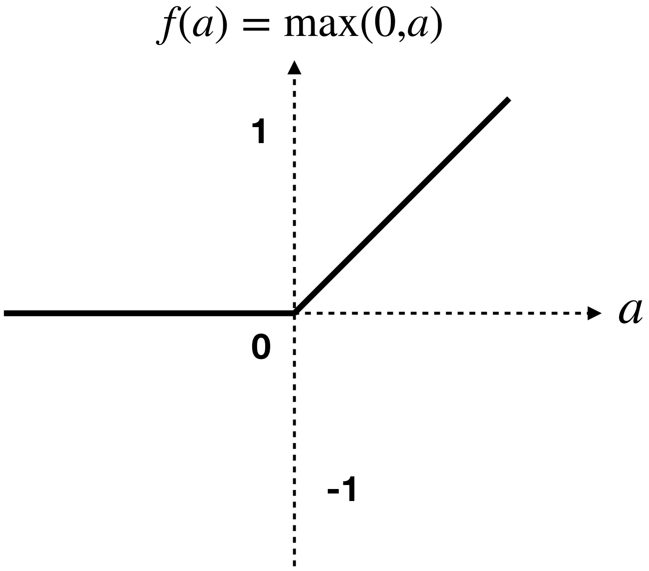


Figure 2.7. ReLU activation function.

networks. As such, it combines the outputs of its previous layer and turns them into vectors.

Apart from the layers mentioned above, the activation functions are also of crucial importance in deep CNNs. They introduce non-linearities to the systems, enabling them to model a diverse range of functions. The ReLU is among the most widely used activation functions in deep learning, which is defined as $f(a) = \max(0, a)$. The plot of the ReLU function is shown in Figure 2.7. It makes the deep neural networks easy to optimize by the gradient-based methods since it is nearly linear [58, 59]. That explains why it becomes the first choice of most deep neural networks instead of sigmoid or tanh functions [56]. However, the ReLU activation function may cause some neurons never to be activated when large gradient flows through a ReLU neuron. The above problem named "dying ReLU" has been relieved by the leaky ReLU function introduced in [60].

To pursue the improvement in performance, CNNs are usually constructed in various architectures using the layers and activation functions mentioned above (sometimes together with other types of layers as well). Some architectures played important roles in the development of CNNs, including the LeNet [61], AlexNet [62], GoogLeNet [63–65], VGGNet [66], and Resnet [67]. The GoogleNet and Resnet are still among the state-of-the-art models which achieve good classification performance. The Resnet and SqueezeNet will be introduced subsequently in details.

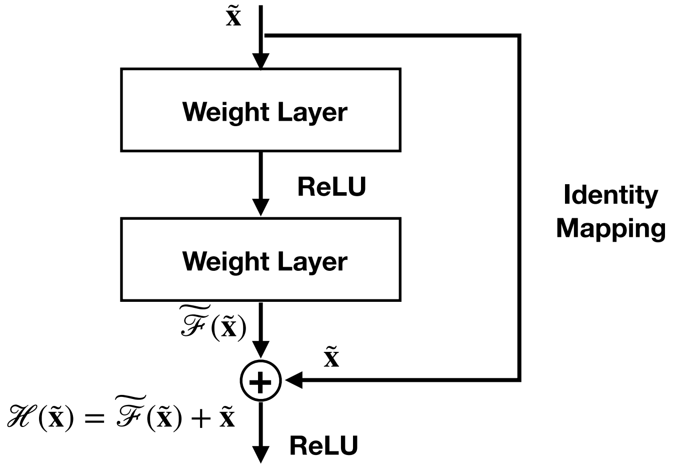

$\tilde{\mathbf{x}}$
Weight Layer
ReLU
Weight Layer
Identity
Mapping
$\widetilde{\mathscr{F}}(\tilde{\mathbf{x}})$
$\tilde{\mathbf{x}}$
$\mathscr{H}(\tilde{\mathbf{x}}) = \widetilde{\mathscr{F}}(\tilde{\mathbf{x}}) + \tilde{\mathbf{x}}$
ReLU

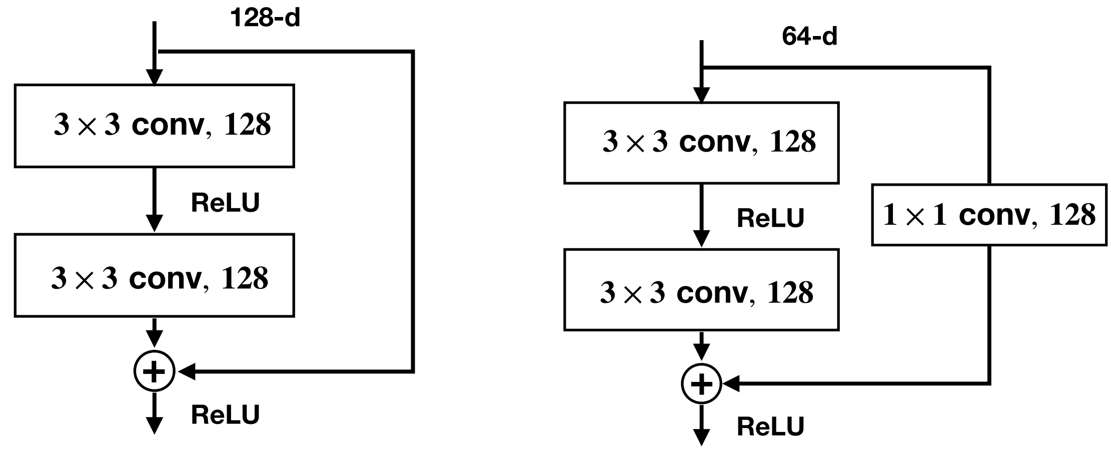
Figure 2.8. Schematic diagram of a building block of residual learning.

2.6.2 Deep Residual Learning

The depth of deep neural networks is crucial for various tasks, especially computer vision related ones. Deep and very deep CNNs have achieved significant success in many tasks. However, accuracy does not always increase with the number of layers. The increasing depth of networks has brought the problem of vanishing or exploding gradients, which has been greatly alleviated by normalization layers. Besides, another problem of degradation appears, i.e., the accuracy of a network gets saturated or decreases with an increasing number of layers. In order to solve this problem, a framework of residual learning is proposed for deep CNNs. Conventional CNNs employ layers to fit the non-linear mapping $\mathcal{H}(\tilde{\mathbf{x}})$ from the input $\tilde{\mathbf{x}}$ to the output of these layers. In contrast, the framework of residual learning fits the residual mapping (residual function) with layers. The residual function is defined as $\widetilde{\mathcal{F}}(\tilde{\mathbf{x}}) = \mathcal{H}(\tilde{\mathbf{x}}) - \tilde{\mathbf{x}}$, of which a diagram is shown in Figure 2.8. As such, the original mapping can be represented by $\widetilde{\mathcal{F}}(\tilde{\mathbf{x}}) + \tilde{\mathbf{x}}$, where the symbol '+' here denotes element-wise addition. It is believed that optimizing a residual mapping is easier than optimizing the original mapping.

The element-wise addition of $\widetilde{\mathcal{F}}(\tilde{\mathbf{x}})$ and $\tilde{\mathbf{x}}$ can be implemented by an identity shortcut connection if the input and the output are of the same dimension, of which an example is shown in Figure 2.9a. When the dimensions are different, a projection shortcut connection is placed that is implemented by a 1×1 convolutional layer (Figure 2.10a).

The results in [67] show the effectiveness of the building blocks for residual learning. For very deep CNNs, a bottleneck building block is proposed to reduce the computational complexity. Example blocks with identity and projection connections are illustrated in Figure 2.10a and Figure 2.10b, respectively. The bottleneck block includes three convolutional layers of 1×1, 3×3, and 1×1 filters. The first 1×1



(a) Building block for the layers of which the input and output are of the same dimensions.

(b) Building block for the layers of increasing dimensions.

Figure 2.9. Building block of a deep residual function for visual recognition tasks.

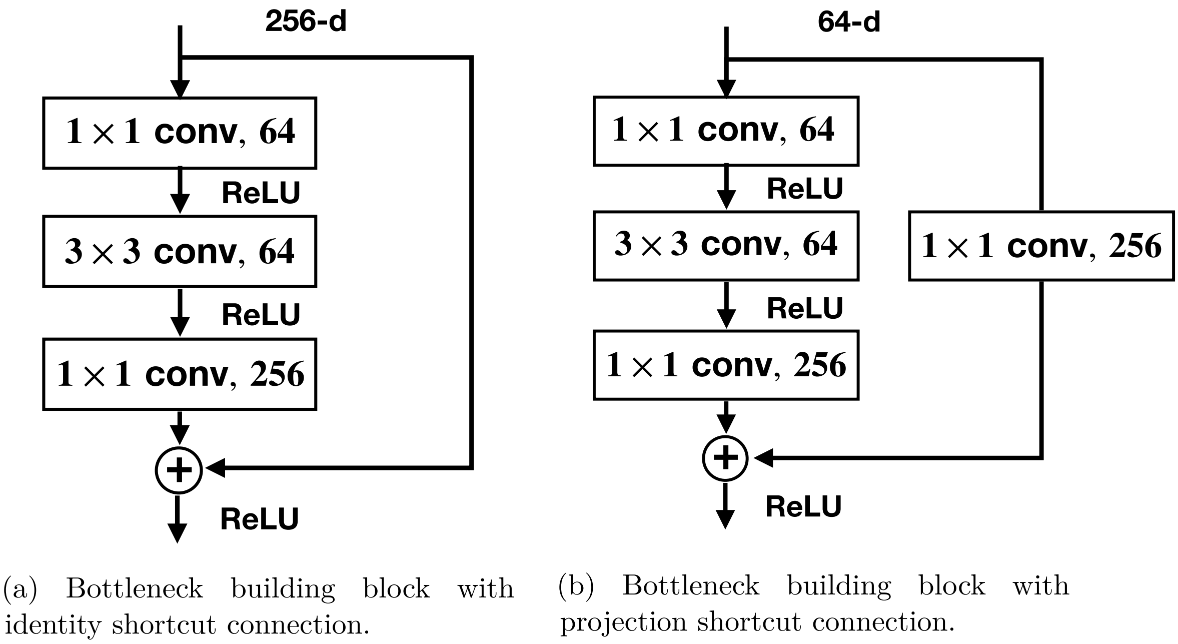
layer reduces the input dimension of the 3 × 3 layer while the other 1 × 1 layer is used to restore the output dimension. The bottleneck block is proposed only for reducing the computational burden.

The architecture of the Resnet50 (50 layers) is shown in Figure 2.11. It starts with a 7 × 7 convolutional layer with a stride of 2, of which the dimension of output is 64. Four bottleneck building blocks are repeated in the network. The downsampling operation with a stride of 2 is conducted by the first blocks of the repeated Bottleneck2, Bottleneck3, and Bottleneck4 blocks, i.e., the Bottleneck2_1, Bottleneck3_1, and Bottleneck4_1, respectively. Batch normalization is conducted after each convolutional layer before the ReLU activation.

The flame detection frameworks based on flame R-CNN and faster R-CNN proposed in Chapter 5 are fine tuned on a Resnet50 trained on the ImageNet database [68].

2.6.3 SqueezeNet

The SqueezeNet [39] is an architecture of deep CNNs which has a smaller number of parameters and similar accuracy compared with the Alexnet [62]. Instead of improving the accuracy, the SqueezeNet focuses on reducing the number of parameters, which will contribute to small-sized CNNs. CNNs of small sizes can be deployed to hardware with limited memory, or exported to hardware from the cloud in real-time. The SqueezeNet designs its architecture according to three strategies:



(a) Bottleneck building block with identity shortcut connection.

(b) Bottleneck building block with projection shortcut connection.

Figure 2.10. Bottleneck architecture of a deep residual function.

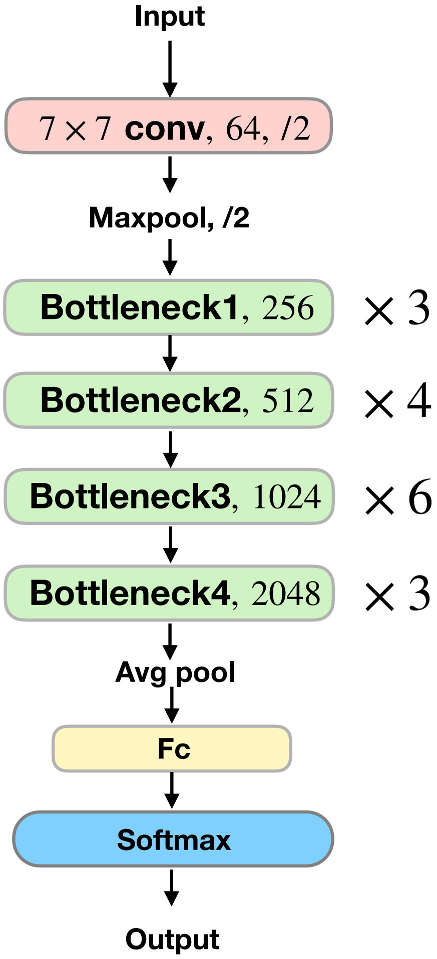


Figure 2.11. Schematic representation of the architecture of Resnet50.

- **Use 1×1 filters instead of some of the 3×3 filters.** Since the parameters of a 1×1 filter are 9X fewer than those of a 3×3 filter, the SqueezeNet replaces some of the 3×3 filters with 1×1 filters.

- **Reduce the input channels of 3×3 filters**. This aims at reducing the number of parameters of the CNN as well. Given a layer consisting of 3×3 filters, the number of parameters will be the product of the number of input channels, the number of filters, and $3 * 3$, which increases with the number of input channels.

- **Increase the sizes of activation maps outputted by convolutional layers by downsampling late in the network.** Downsampling can be conducted by convolutional layers or pooling layers in networks. Early downsampling will lead to small activation maps of the layers after downsampling. Usually, larger activation maps contribute to higher accuracy than those of small sizes. Therefore, the downsampling is delayed in the SqueezeNet to maximize accuracy.

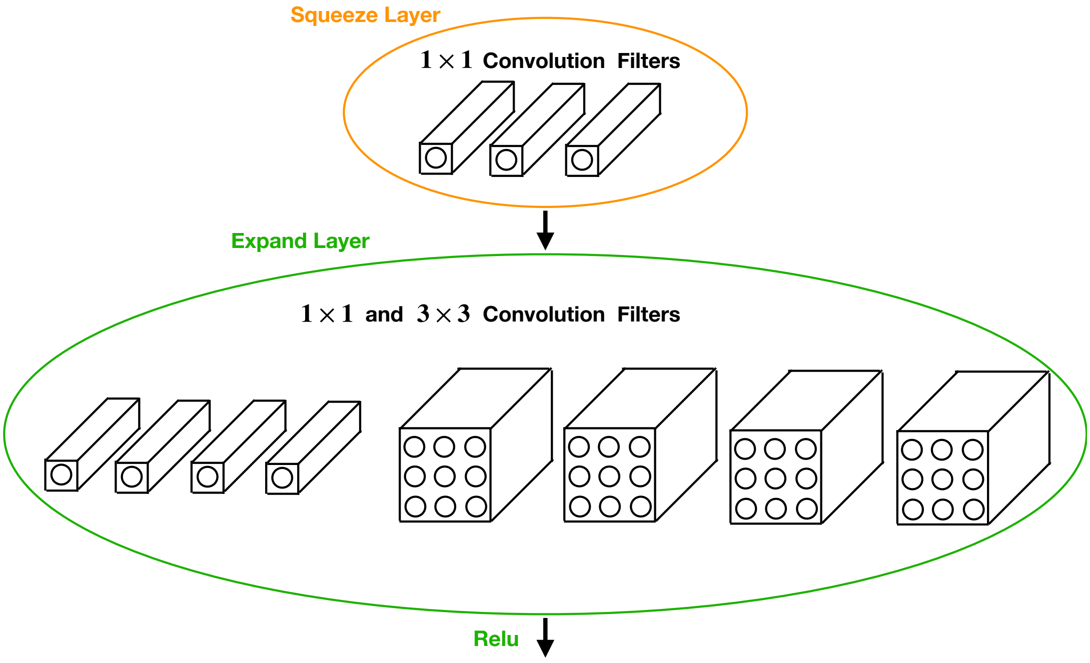

Squeeze Layer
1 × 1 Convolution Filters
Expand Layer
1 × 1 and 3 × 3 Convolution Filters
Relu

Figure 2.12. Schematic representation of the 'fire' module of SqueezeNet.

In the research of deep learning, modules, which are composed of several layers of a fixed structure, are usually defined for convenience. A 'fire' module is proposed in the SqueezeNet according to the first and second strategies mentioned above. It is noteworthy that the ‘fire’ here is not related to physical fires, but is the name of the module. The architecture of the ‘fire’ module is shown in Figure 2.12. It consists of two parts, i.e. a squeeze layer and an expand layer. The squeeze layer is a convolutional layer of only 1 × 1 filters, of which the output is fed into the expand layer. In contrast, the expand layer is comprised of a mix of 1 × 1 and 3 × 3 filters. There are three hyperparameters in a ‘fire’ module, i.e. the number of filters in the squeeze layer, and the numbers of 1 × 1 and 3 × 3 filters in the expand layer, which are denoted by s_{1x1}, e_{1x1}, and e_{3x3}, respectively. The value of s_{1x1} is set to be smaller than $e_{1x1} + e_{3x3}$ to reduce the input channels of 3 × 3 filters as suggested by the second strategy. Additionally, the expand layer replaces some of the 3 × 3 filters with 1 × 1 filters, which is suggested by the first strategy.

The architecture of the SqueezeNet is illustrated in Figure 2.13. The one on the left shows a simple version of the SqueezeNet (vanilla SqueezeNet), while the figures in the middle and on the right show the SqueezeNet with simple and complex bypass connections, respectively. It can be seen that eight ‘fire’ modules are placed after a convolutional layer, and feed the output to a final convolutional layer. The number of filters is increased gradually from the beginning to the end of the network. Specifically, the numbers of filters in expand layers are increased every two modules by 128, as shown in Figure 2.13. The number of filters in a squeeze layer is proportional to that in the expand layer of the same ‘fire’ module, of which the ratio is defined as a hyperparameter called squeeze ratio. The squeeze ratio controls the number of the input channels to be fed into 3 × 3 filters. According to the third strategy discussed above, the squeeze ratio should be set to a relatively small value. Besides, another hyperparameter is proposed to control the percentage of 3 × 3 filers in each expand layer, which is shared by all ‘fire’ modules in the same network. Furthermore, three max-pooling layers are placed after conv1, fire4 and fire8 with a stride of 2, which are relatively late downsampling operations in the network. ReLU is applied to both squeeze and expand layers, and dropout of 50% is placed after the module ‘fire9’.

In the architecture using simple bypass connections, the ‘fire’ modules with simple shortcut connections around them need to learn a residual function between the input and the output, and the input of the modules next to them will be changed accordingly. For example, the input of the module ‘Fire8’ is the element-wise addition of the output of the modules ‘Fire6’ and ‘Fire7’. It is noteworthy that simple bypass connections will not increase the number of parameters of the network. However, the simple connections can be placed only around the modules of which the number of input channels is the same as that of the output channels. To solve this problem, complex connections are proposed using convolutional layers of 1 × 1 filters to change the dimension of the input of some ‘fire’ modules, which will increase the number of parameters. According to the results in [39], both simple and complex shortcut connections improve the accuracy of the SqueezeNet compared with the vanilla one. However, the network using simple bypass connections achieves better performance than that using complex connections, which is different from expected.

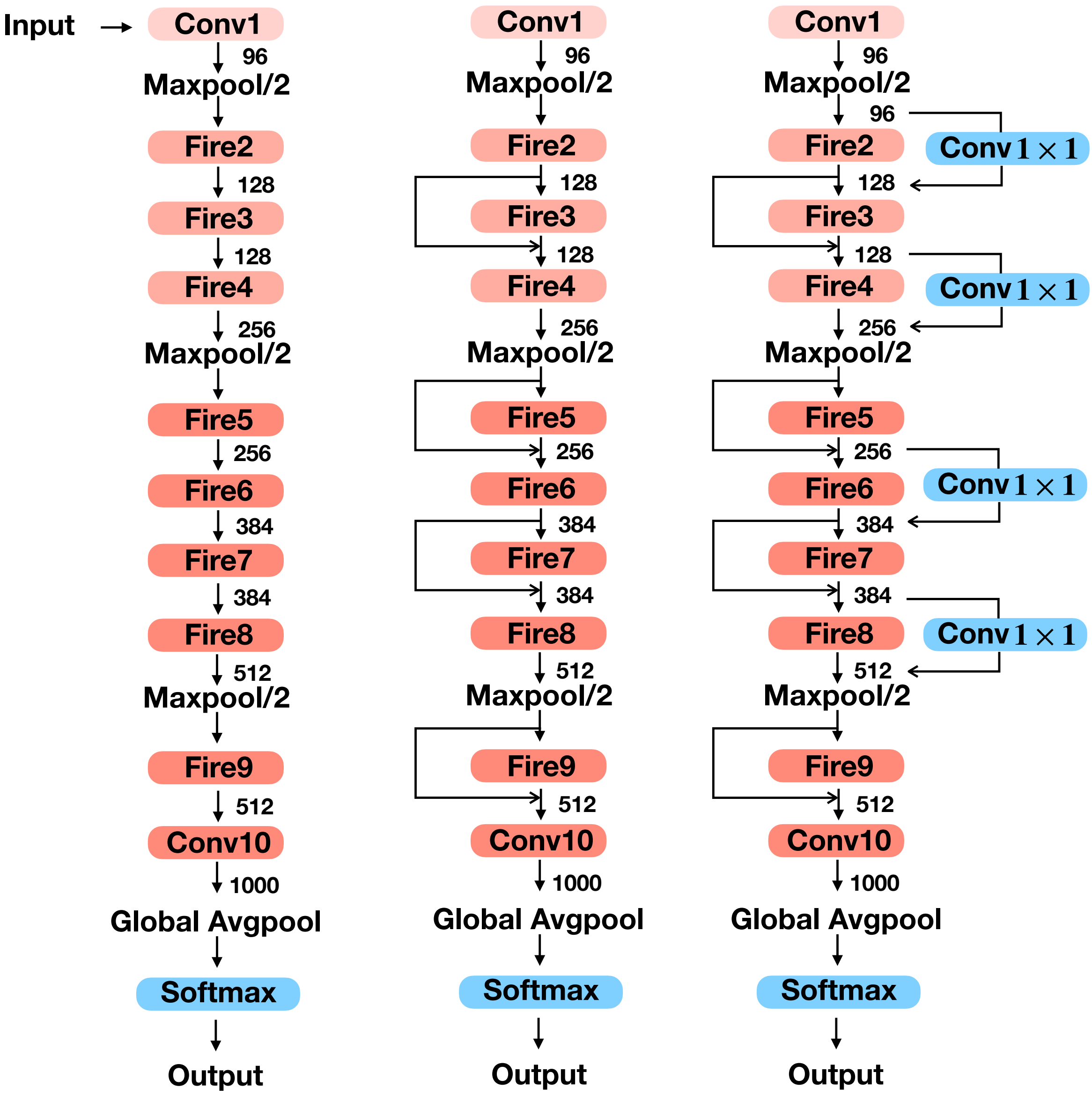


Figure 2.13. The architecture of the SqueezeNet. left: vanilla SqueezeNet; middle: SqueezeNet with simple bypass connections; right: SqueezeNet with complex bypass connections.

A framework based on the vanilla SqueezeNet is proposed for flame detection in [2]. Its results are compared with the proposed frameworks in Chapter 3, Chapter 4, and Chapter 5 of this thesis. In the framework for comparison, the squeeze ratio and percentage of 3 × 3 filters of each module are set to 0.125 and 50%, respectively.

Chapter 3

Autonomous Flame Detection Based on Optical Flow and Saliency Analysis

The diverse appearance and rapid changes of flames make it challenging to achieve accurate detection, as mentioned in Section 2.1.1. To overcome these difficulties, a hybrid framework is proposed in this chapter based on a probabilistic saliency analysis approach, an optical flow estimation algorithm and a temporal wavelet transform based analysis scheme. The motion of objects in each frame is estimated by the Horn-Schunck optical flow algorithm. Subsequently, a saliency map is obtained with a probabilistic saliency analysis method based on the intensity values and the magnitudes of the estimated optical flows to select regions with dramatic motion and high brightness. Additionally, after processing the saliency map with a group of chromatic rules of flames, the proposed framework describes the quasi-periodic behaviour of the pixels in flame boundaries with the features extracted by a temporal wavelet transform based analysis which reduces the false alarm rate significantly. The probabilistic saliency analysis combines the dynamic and bright characteristics of flames which contributes to accurate detection of flames together with the selective rules of colours and temporal wavelet transform based analysis.

3.1 The Framework of the Proposed Method

The proposed framework of flame detection is based on the combination of a probabilistic saliency analysis method, the Horn-Schunck optical flow estimation algorithm, several chromatic selective rules of flames, and a temporal wavelet

The work in this chapter is published in "Zhenglin Li, Olga Isupova, Lyudmila Mihaylova, and Lucile Rossi, "Autonomous flame detection in video based on saliency analysis and optical flow," in *Proc. Int. Conf. Multisensor Fusion and Integration*, 2016, pp. 218-223".

transform based analysis approach. It fuses different features extracted by several sub-phases of the framework to describe the properties of flames. Based on the hybrid system, candidate flame pixels of each frame can be detected accurately to provide reliable frame-wise results.

The flow diagram is shown in Figure 3.1 to illustrate the proposed framework for flame detection. The optical flows are estimated by the Horn–Schunck algorithm, which is illustrated in Section 3.2. The magnitudes of the estimated optical flows instead of the orientations are set as features and will be further processed to generate a motion saliency map, which helps to select probable flame regions based on the dynamic properties of flames. Additionally, another saliency map with intensity values as features is combined with the optical flow based map by averaging. As such, both the dynamic and bright characteristics of flames are taken into account which can help to detect candidate flame regions for further processing.

Subsequently, the combined saliency map is further processed by several flame colour selective rules, to discard pixels of different colours from flames. Specifically, rules in (2.19)-(2.24) are selected because of their robustness to the changes of illumination [4]. Saliency values of the pixels filtered out in this phase are set to zero, which reduces the computational burden of the framework. Afterwards, binarization is performed on the processed map by setting a threshold for the saliency values. In order to reduce the influence of noises, a morphological operation called closing [69] is conducted on the binary map.

The temporal wavelet transform based analysis introduced in Section 3.4 is the final step of the proposed framework. Pixels not satisfying the constraints are classified as non-flame ones. A frame-wise decision on the existence of flames can be made based on the number of detected flame pixels, which can give a warning or even trigger fire alarms once flames are detected.

3.2 Horn–Schunck Optical Flow Estimation Algorithm

The framework estimates the optical flows using the Horn–Schunck algorithm [15] to generate the motion saliency map which is used for detecting flames based on the dynamic property. Let $V(x_c, y_c, t_c)$ denote the intensity value of a point located at (x_c, y_c) in the frame at time t_c, which can reflect the brightness of the point. It is assumed that the intensity V is continuous and differentiable, and the measured intensity of pixels in videos are discrete values sampled from the intensity field. Similarly, the coordinates (x_c, y_c) and time t_c are treated as continuous variables in derivation, and in practice, they are sampled on a grid at regular intervals.

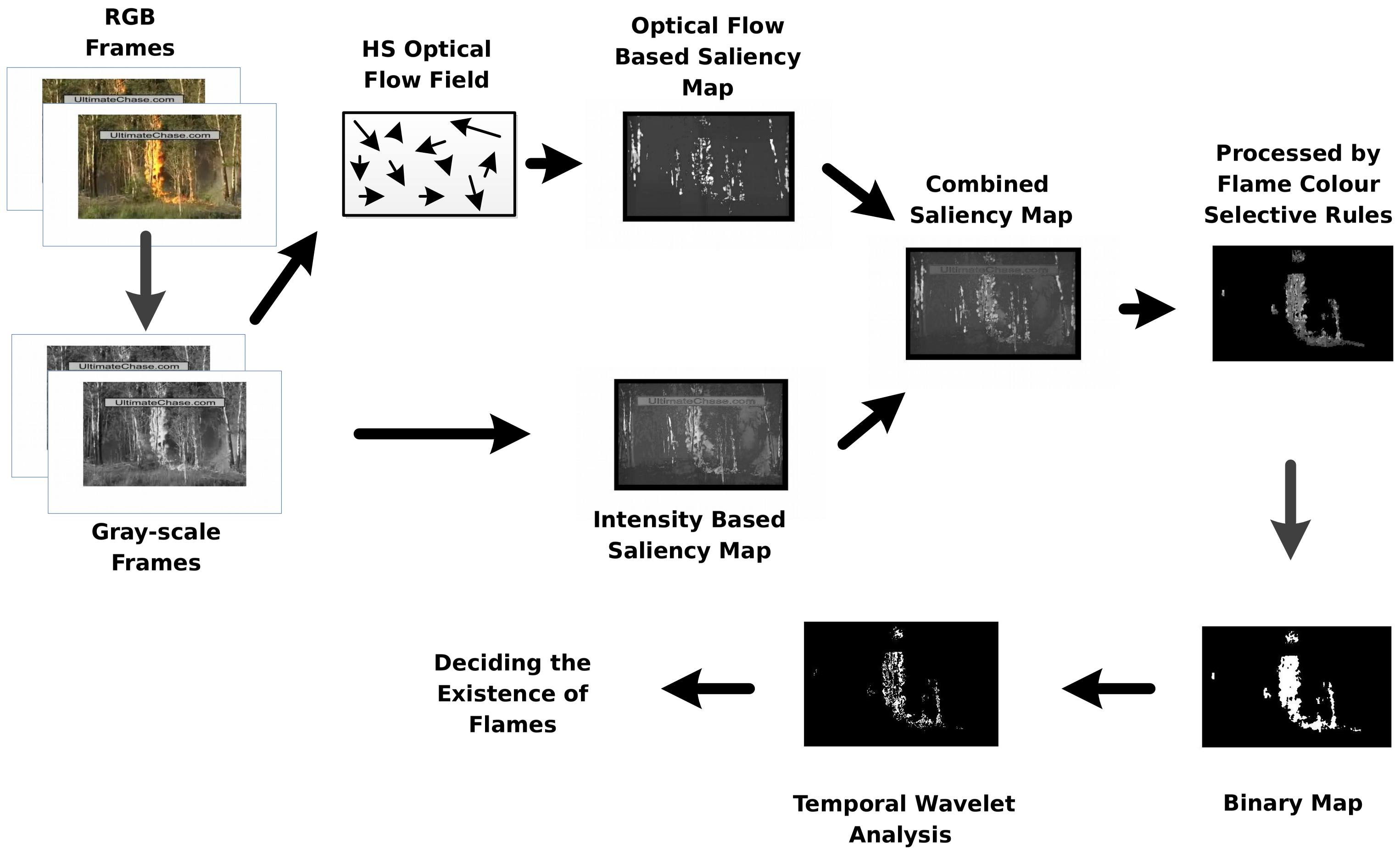

RGB Frames
UltimateChase.com
HS Optical Flow Field
Optical Flow Based Saliency Map
Combined Saliency Map
Processed by Flame Colour Selective Rules
Gray-scale Frames
Intensity Based Saliency Map
Deciding the Existence of Flames
Temporal Wavelet Analysis
Binary Map

Figure 3.1. The diagram of the proposed framework based on optical flow and saliency analysis.

The algorithm assumes that the brightness of a point across several frames is constant, so the constraint below needs to be satisfied

$$V(x_c + \delta x_c, y_c + \delta y_c, t_c + \delta t_c) = V(x_c, y_c, t_c), \tag{3.1}$$

where $\delta x_c, \delta y_c$, and δt_c represent the displacement in the direction of x, y, and t.

Linearisation of Eq. (3.1) using the first-order Taylor expansion yields the optical flow constraint equation, which is given by

$$\dot{V}_x u + \dot{V}_y v + \dot{V}_t = 0, \tag{3.2}$$

where $\dot{V}_x = \partial V/\partial x_c$, $\dot{V}_y = \partial V/\partial y_c$, $\dot{V}_t = \partial V/\partial t_c$, and the optical flow velocity (u, v) is defined as $u = dx_c/dt_c$, $v = dy_c/dt_c$.

As two unknown variables cannot be obtained from one equation, an additional constraint, known as the smoothness constraint, is introduced. It assumes that the velocities of neighbouring points are similar and the velocity field varies smoothly almost everywhere. A measure of smoothness is defined as:

$$\|\nabla u\|_2^2 + \|\nabla v\|_2^2 = \left(\frac{\partial u}{\partial x_c}\right)^2 + \left(\frac{\partial u}{\partial y_c}\right)^2 + \left(\frac{\partial v}{\partial x_c}\right)^2 + \left(\frac{\partial v}{\partial y_c}\right)^2, \tag{3.3}$$

where ∇ and $\|\cdot\|_2^2$ denote the gradient and the square of L$_2$ norm, respectively.

The combination of the optical flow constraint and a global smoothness term results in the following penalty function

$$J_{u,v} = \iint [(\dot{V}_x u + \dot{V}_y v + \dot{V}_t)^2 + \alpha^2(\|\nabla u\|_2^2 + \|\nabla v\|_2^2)]dx_c\, dy_c, \tag{3.4}$$

where the parameter α controls the influence of the term of smoothness. As the brightness constancy constraint in Eq. (3.1) is not always satisfied because of the changes of illumination or existence of noise, the penalty function aims at minimizing the errors (or in other words, the disorders) as well as obtaining better smoothness.

To minimize the penalty function in Eq. (3.4), the Euler-Lagrange equation [70] is employed and the optimal optical flow should satisfy

$$\dot{V}_x^2 u + \dot{V}_x \dot{V}_y v + \dot{V}_x \dot{V}_t - \alpha^2 \nabla^2 u = 0, \tag{3.5}$$

$$\dot{V}_x \dot{V}_y u + \dot{V}_y^2 v + \dot{V}_y \dot{V}_t - \alpha^2 \nabla^2 v = 0, \tag{3.6}$$

where $\nabla^2 u$ and $\nabla^2 v$ denote the Laplacians of u and v, respectively.

As the measured intensity values are taken at discrete spatial and temporal coordinates, the partial derivatives $\dot{V}_x, \dot{V}_y, \dot{V}_t$ and the Laplacians $\nabla^2 u$ and $\nabla^2 v$ need to be approximated. Denote the measured intensity value of the point located at the

y_s-th row and x_s-th column of the t-th frame as $V(x_s, y_s, t)$. The partial derivatives are approximated as the average of the first-order differences of adjacent pixels as

$$\begin{aligned}\dot{V}_x =& \frac{1}{4}\Big\{V(x_s+1, y_s, t) - V(x_s, y_s, t) + V(x_s+1, y_s+1, t) - V(x_s, y_s+1, t) + \\ & V(x_s+1, y_s, t+1) - V(x_s, y_s, t+1) + V(x_s+1, y_s+1, t+1) - V(x_s, y_s+1, t+1)\Big\},\end{aligned} \tag{3.7}$$

$$\begin{aligned}\dot{V}_y =& \frac{1}{4}\Big\{V(x_s, y_s+1, t) - V(x_s, y_s, t) + V(x_s+1, y_s+1, t) - V(x_s+1, y_s, t) + \\ & V(x_s, y_s+1, t+1) - V(x_s, y_s, t+1) + V(x_s+1, y_s+1, t+1) - V(x_s+1, y_s, t+1)\Big\},\end{aligned} \tag{3.8}$$

$$\begin{aligned}\dot{V}_t =& \frac{1}{4}\Big\{V(x_s, y_s, t+1) - V(x_s, y_s, t) + V(x_s+1, y_s, t+1) - V(x_s+1, y_s, t) + \\ & V(x_s, y_s+1, t+1) - V(x_s, y_s+1, t) + V(x_s+1, y_s+1, t+1) - V(x_s+1, y_s+1, t)\Big\}.\end{aligned} \tag{3.9}$$

The Laplacians are estimated as

$$\nabla^2 u = \kappa\Big\{\bar{u}(x_s, y_s, t) - u(x_s, y_s, t)\Big\} \tag{3.10}$$

$$\nabla^2 v = \kappa\Big\{\bar{v}(x_s, y_s, t) - v(x_s, y_s, t)\Big\} \tag{3.11}$$

where

$$\begin{aligned}\bar{u}(x_s, y_s, t) =& \frac{1}{6}\Big\{u(x_s-1, y_s, t) + u(x_s, y_s-1, t) + u(x_s+1, y_s, t) + u(x_s, y_s+1, t)\Big\} \\ +& \frac{1}{12}\Big\{u(x_s-1, y_s-1, t) + u(x_s+1, y_s-1, t) + u(x_s-1, y_s+1, t) + u(x_s+1, y_s+1, t)\Big\},\end{aligned} \tag{3.12}$$

$$\begin{aligned}\bar{v}(x_s, y_s, t) =& \frac{1}{6}\Big\{v(x_s-1, y_s, t) + v(x_s, y_s-1, t) + v(x_s+1, y_s, t) + v(x_s, y_s+1, t)\Big\} \\ +& \frac{1}{12}\Big\{v(x_s-1, y_s-1, t) + v(x_s+1, y_s-1, t) + v(x_s-1, y_s+1, t) + v(x_s+1, y_s+1, t)\Big\}.\end{aligned} \tag{3.13}$$

Eqs. (3.5) and (3.6) can be represented in an alternative form by substituting the Laplacians with the approximated ones in Eqs. (3.10) and (3.11), i.e.,

$$(\alpha^2 + \dot{V}_x^2)u + \dot{V}_x\dot{V}_y v = (\alpha^2\bar{u} - \dot{V}_x\dot{V}_t) \tag{3.14}$$

$$\dot{V}_x\dot{V}_y u + (\alpha^2 + \dot{V}_x^2)v = (\alpha^2\bar{v} - \dot{V}_y\dot{V}_t). \tag{3.15}$$

Solving the equations above yields

$$u = \bar{u} - \frac{\dot{V}_x[\dot{V}_x\bar{u} + \dot{V}_y\bar{v} + \dot{V}_t]}{\alpha^2 + \dot{V}_x^2 + \dot{V}_y^2}, \tag{3.16}$$

$$v = \bar{v} - \frac{\dot{V}_y[\dot{V}_x\bar{u} + \dot{V}_y\bar{v} + \dot{V}_t]}{\alpha^2 + \dot{V}_x^2 + \dot{V}_y^2}, \tag{3.17}$$

based on which the numerical estimation of the optical flow can be calculated in an iterative way.

The magnitude $\sqrt{u^2 + v^2}$ of the estimated optical flow velocity (u, v) can describe the motion of points in frames. It is expected that the points in flame regions have larger magnitudes of optical flows than those belonging to the background since flames usually change rapidly and wildly. However, the magnitudes cannot be used to select points with large motion directly by being compared with a predetermined threshold, because the estimated magnitudes of distant targets are usually much smaller than those of objects near cameras. A preset threshold may result in either high false alarm rates of videos, in which objects move near cameras, or low true positive rates of videos containing distant flames. Therefore, regions with relatively larger motion compared with surroundings should be detected for robust detection of flames. To accomplish this goal, the proposed framework selects regions whose magnitudes of optical flows are larger than the pixels around, using a probabilistic saliency analysis approach which will be introduced in Section 3.3.

3.3 Probabilistic Flame Saliency Analysis

The probabilistic saliency analysis approach aims at measuring the saliency of pixels based on the semi-local feature contrast [71, 72]. Specifically, the motion saliency map assigns high values to the pixels of which the magnitudes of optical flows are salient in a frame. Thus the regions with larger motion than surroundings can be selected based on this saliency map. Similarly, an intensity saliency map is also employed for detecting candidate flames because regions of flames are usually brighter than the background. The obtained saliency maps can measure the probability of a pixel being part of flames.

A sliding rectangular window is needed in the approach, which contains two parts, the inner kernel and the border, as shown in Figure 3.2.

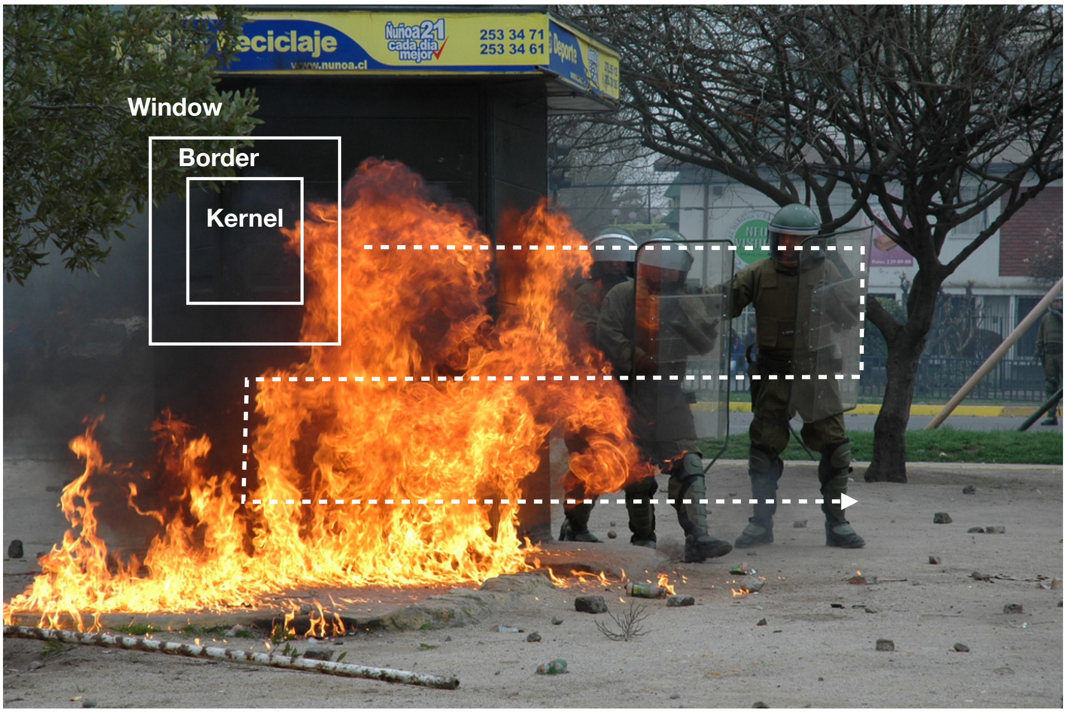


Figure 3.2. A schematic diagram of the sliding window for saliency analysis.

The widths and heights of the window and kernel are represented by w_W, h_W, and w_K, h_K, respectively. Let $F(x_s, y_s)$ denote the feature of the point located at (x_s, y_s). In this chapter, the intensity value and the magnitude of the estimated optical flow of each pixel are used as features to produce two saliency maps.

Two hypotheses are proposed as follows

$$\mathcal{H}_0 : \text{the point is not salient,} \tag{3.18}$$

$$\mathcal{H}_1 : \text{the point is salient.} \tag{3.19}$$

The prior probabilities are represented by $p(\mathcal{H}_0)$ and $p(\mathcal{H}_1)$, respectively. It is first assumed that the hypothesis $\mathcal{H}_0$ is valid when a pixel is located in the area of border, while the hypothesis $\mathcal{H}_1$ corresponds to the situation that a pixel is inside the kernel area. The prior probabilities satisfy $p(\mathcal{H}_0) = 1 - p(\mathcal{H}_1)$. It is reasonable to calculate them according to the area ratios of the kernel and border parts. The posterior $p(\mathcal{H}_1|F(x_s, y_s))$ reflects the probability of the point at (x_s, y_s) being salient based on its feature $F(x_s, y_s)$, so the saliency $\boldsymbol{\zeta}(x_s, y_s)$ is defined as

$$\boldsymbol{\zeta}(x_s, y_s) = p\left(\mathcal{H}_1 | F(x_s, y_s)\right). \tag{3.20}$$

Using the Bayes' theorem

$$p(\mathcal{H}_1 | F(x_s, y_s)) = \frac{p(\mathcal{H}_1) p(F(x_s, y_s) | \mathcal{H}_1)}{p(F(x_s, y_s))}, \tag{3.21}$$

where $p(F(x_s, y_s))$ can be expanded as

$$p(F(x_s, y_s)) = p(\mathcal{H}_1) p(F(x_s, y_s) | \mathcal{H}_1) + p(\mathcal{H}_0) p(F(x_s, y_s) | \mathcal{H}_0). \tag{3.22}$$

Moreover, the likelihood $p(F(x_s, y_s)|\mathcal{H}_1)$ and $p(F(x_s, y_s)|\mathcal{H}_0)$ can be estimated using histograms of feature $F(\cdot, \cdot)$ computed in the kernel and border areas of the sliding window at each location. The histograms are denoted by $hist_K(F)$ and $hist_B(F)$, respectively. To enhance the robustness, the obtained histograms are smoothed by a Gaussian blurring function before normalization.

$$\hat{p}(F(x_s, y_s) | \mathcal{H}_1) = Norm\left(\widetilde{g}(F) * hist_K(F)\right), \tag{3.23}$$

$$\hat{p}(F(x_s, y_s) | \mathcal{H}_0) = Norm\left(\widetilde{g}(F) * hist_B(F)\right), \tag{3.24}$$

where $\hat{p}(F(x_s, y_s)|\mathcal{H}_1)$ and $\hat{p}(F(x_s, y_s)|\mathcal{H}_0)$ are the estimated probabilities, $*$ denotes the operation of convolution, $\widetilde{g}(F)$ represents the employed Gaussian blurring function and $Norm(\cdot)$ denotes the histogram normalization operation which is defined as

$$Norm(h(i)) = \frac{1}{\sum_i h(i)} h(i). \tag{3.25}$$

The sliding window located at the j-th position is denoted by $\widetilde{W}(j)$. When it slides with a step s_W, windows at different positions may overlap each other. If it happens, the saliency value $\boldsymbol{\zeta}(x_s, y_s)$ of a point located in the overlapping area of several windows is calculated as

$$\boldsymbol{\zeta}(x_s, y_s) = \max_j \left\{ \boldsymbol{\zeta}_j(x_s, y_s) | (x_s, y_s) \in \widetilde{W}(j) \right\}. \tag{3.26}$$

Multi-scaled steps and windows are employed to reduce the influence of the sizes of steps and windows.

Using the saliency estimation approach mentioned above, two saliency maps are obtained by setting the intensity values and the optical flow magnitudes as features, respectively. Subsequently, these two maps are averaged with equal weights, which takes into account both the dynamic and bright characteristics of flames. An example frame and its saliency maps are shown in Figure 3.6. It can be seen that the flame regions have been assigned high values in the combined saliency map, while other parts of the frame without flames have low saliency values. The saliency map can effectively select probable flame pixels, and thus reduces the errors of final detection.

3.4 Temporal Wavelet Transform Based Analysis

Different from the movement of most objects, flames usually change dramatically with time. Research indicates that turbulent flames flicker at frequencies around 10Hz [5]. To distinguish flames from other moving objects based on this flickering property, the temporal wavelet transform based analysis approach is introduced to the framework. The method applies a two-stage filter bank based on the one-dimensional discrete wavelet transform. The extracted high-frequency sub-signals can reflect the difference in the temporally changing patterns between the pixels of flames and non-flame objects.

Denote $V_R(x_s, y_s, t)$ as the intensity value of the R channel of the pixel located at (x_s, y_s) in the t-th frame. A temporal series of the R values can be represented as $V_R(x_s, y_s, t : t+T-1) = [V_R(x_s, y_s, t), V_R(x_s, y_s, t+1), \cdots, V_R(x_s, y_s, t+T-1)]$, with the length of T. The two-stage filter bank for analysing the temporal changes of candidate flame pixels is shown in Figure 3.3.

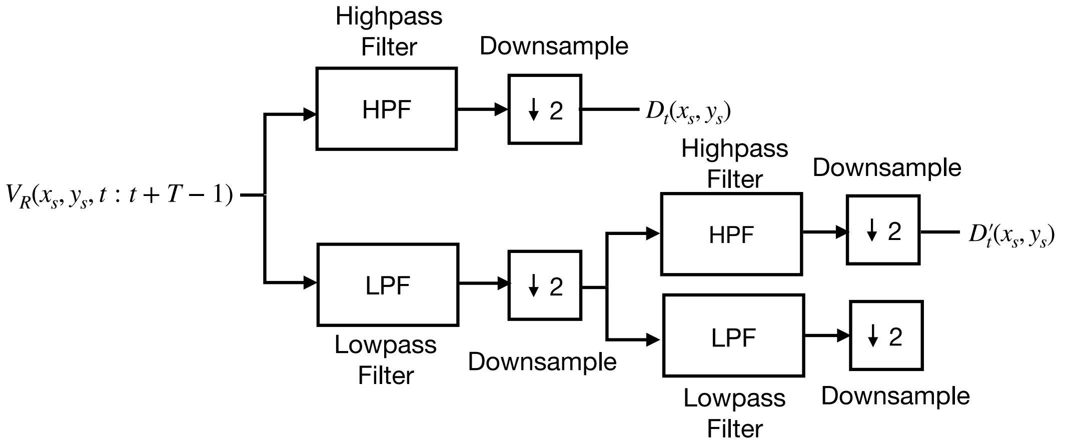
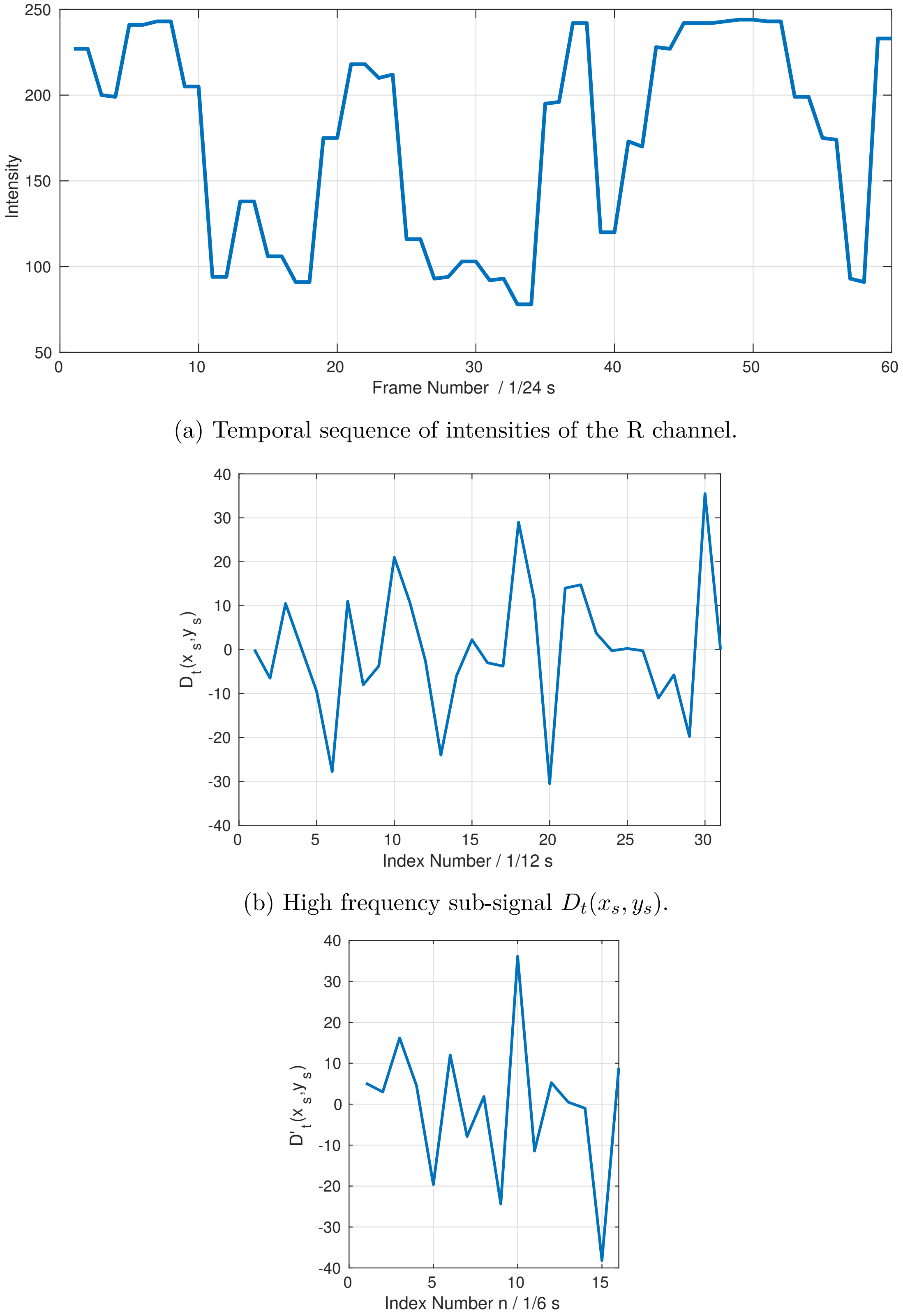


Figure 3.3. The two-stage filter bank based on one-dimensional wavelet transform.

The half-band high-pass and low-pass filters employed in the approach are with coefficients of [-0.25, 0.5, -0.25] and [0.25, 0.5, 0.25], respectively. The high-frequency sub-signals $D_t(x_s, y_s)$ and $D_t'(x_s, y_s)$ are employed to distinguish flame pixels from the pixels part of moving objects or stationary background.

For a pixel belonging to the background, $D_t(x_s, y_s)$ and $D_t'(x_s, y_s)$ are around zero because of the absence of high-frequency activities. By contrast, the two sub-signal

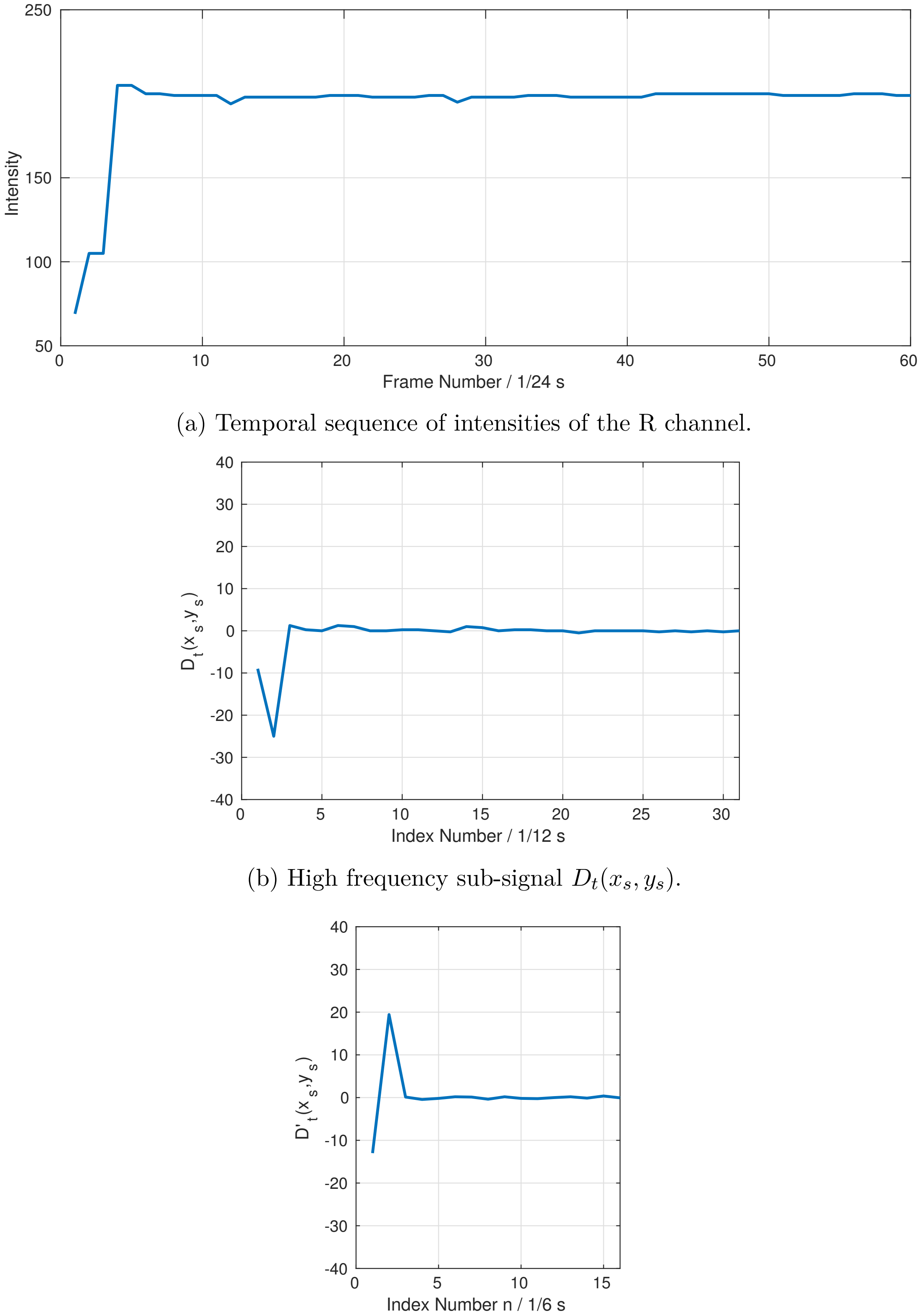


(a) Temporal sequence of intensities of the R channel.

(b) High frequency sub-signal $D_t(x_s, y_s)$.

(c) Higher part $D_t'(x_s, y_s)$ of low frequency sub-signal.

Figure 3.4. Temporal variation of the intensity values of the R channel of a flame pixel.



(a) Temporal sequence of intensities of the R channel.

(b) High frequency sub-signal $D_t(x_s, y_s)$.

(c) Higher part $D_t'(x_s, y_s)$ of low frequency sub-signal.

Figure 3.5. Temporal variation of the intensity values of the R channel of a flame-coloured pixel which is part of a non-flame moving object.

series of a pixel part of a moving flame-coloured object have one or two spikes, while the curves of flame pixels, especially those near the boundaries of flames, have several spikes resulted from the flickering of turbulent flames. Figure 3.4 and Figure 3.5 show the differences between the high-frequency sub-signals of pixels part of a flame and a moving object.

3.5 Experiments and Discussion

Experiments are carried on colour optical videos to test the performance of the proposed framework. Section 3.5.1 introduces the database of testing videos and the widely accepted evaluation methods briefly. Subsequently, experimental results of the proposed framework are provided in Section 3.5.2 and compared with a state-of-the-art method of flame detection based on the SqueezeNet [2]. Afterwards, the results are discussed and both the advantages and disadvantages of the proposed method are analysed. The influence of the threshold for saliency is also explored by testing the framework with different thresholds.

3.5.1 Benchmarking Database and Performance Evaluation Methods

The proposed framework is tested on 3968 frames from 16 different videos from [73, 74]. The videos for experiments are of various scenes, among which are both positive (containing flames) and negative (containing only moving flame-coloured objects) videos. A brief description of the videos for experiments is presented in Table 3.1.

The performance is evaluated using the true positive rate (TPR) and true negative rate (TNR) [75], which are widely accepted for evaluating frameworks of flame detection. They are defined as

$$TPR = \frac{tp}{tp + fn} \quad , \tag{3.27}$$

$$TNR = \frac{tn}{tn + fp} \quad , \tag{3.28}$$

where *tp* and *tn* denote the numbers of frames with and without flames which are correctly classified, respectively. *fp* represents the number of frames containing no flame but falsely classified as positive ones, while *fn* denotes the number of falsely classified positive frames. The TPR and TNR reflect the sensitivity and specificity of flame detection methods, respectively. High TPRs mean good performance in detecting flames while high TNRs lead to a reduced number of false alarms. However, the TPR and TNR are usually competing, meaning that the improvement of one

Table 3.1. Information of the testing videos for experiments

Video	Burning Objects	Distractors	Positive Frames	Negative Frames	Lighting Condition	Smoke Condition	Location
VC1	Hay	A walking person	26	0	Bright	Thick	Outdoor
VC2	Hay	A working man	93	0	Bright	Thick	Outdoor
VC3	Unknown	Moving people	48	0	Bright	Thick	Outdoor
VC4	Hay	Moving people	41	0	Bright	Thick	Outdoor
VC5	Trees	None	214	0	Bright	Thin	Outdoor
VC6	Trees	None	176	0	Dark	Thin	Outdoor
VC7 [a]	Branches	A walking man	687	5	Bright	Medium	Outdoor
VC8 [b]	Assembly line	Moving workers	572	69	Bright	Thin	Indoor
VC9	Grass	None	386	0	Bright	Medium	Outdoor
VC10	Papers	A moving light	395	0	Bright	Thin	Indoor
VC11	Trees	None	186	0	Bright	Thick	Outdoor
VC12	None	Flashing carlights	0	139	Dark	None	Outdoor
VC13	None	Flashing carlights and walking people	0	144	Dark	None	Outdoor
VC14	None	A walking person in red clothes	0	155	Bright	None	Indoor
VC15	None	Crashing cars	0	378	Bright	None	Indoor
VC16	None	Walking people	0	254	Bright	None	Indoor

[a] Frame 531, 532, 533, 658, 660 are negative, other frames contain flames in them

[b] The first 69 frames are negative and others are positive.

is likely to cause the degeneration of the other. Therefore, flame detection schemes need to balance the sensitivity and specificity to achieve satisfactory performance in both TPR and TNR.

3.5.2 Detection Performance Evaluation and Discussion

3.5.2.1 Experiment Settings

In order to reduce the influence of the size of the sliding window and its stride, multi-scaled windows and steps are employed for robust detection of flames. Sizes of windows and steps used in experiments are as follows

$$w_W = [0.25\ 0.3\ 0.5\ 0.7] \times \max\{w_F, h_F\}, \tag{3.29}$$

$$h_W = [0.1\ 0.3\ 0.5\ 0.4] \times \max\{w_F, h_F\}, \tag{3.30}$$

$$w_K = w_W \times 0.9, \tag{3.31}$$

$$h_K = h_W \times 0.9, \tag{3.32}$$

$$s_W = [0.01\ 0.015\ 0.03\ 0.04] \times \max\{w_F, h_F\}, \tag{3.33}$$

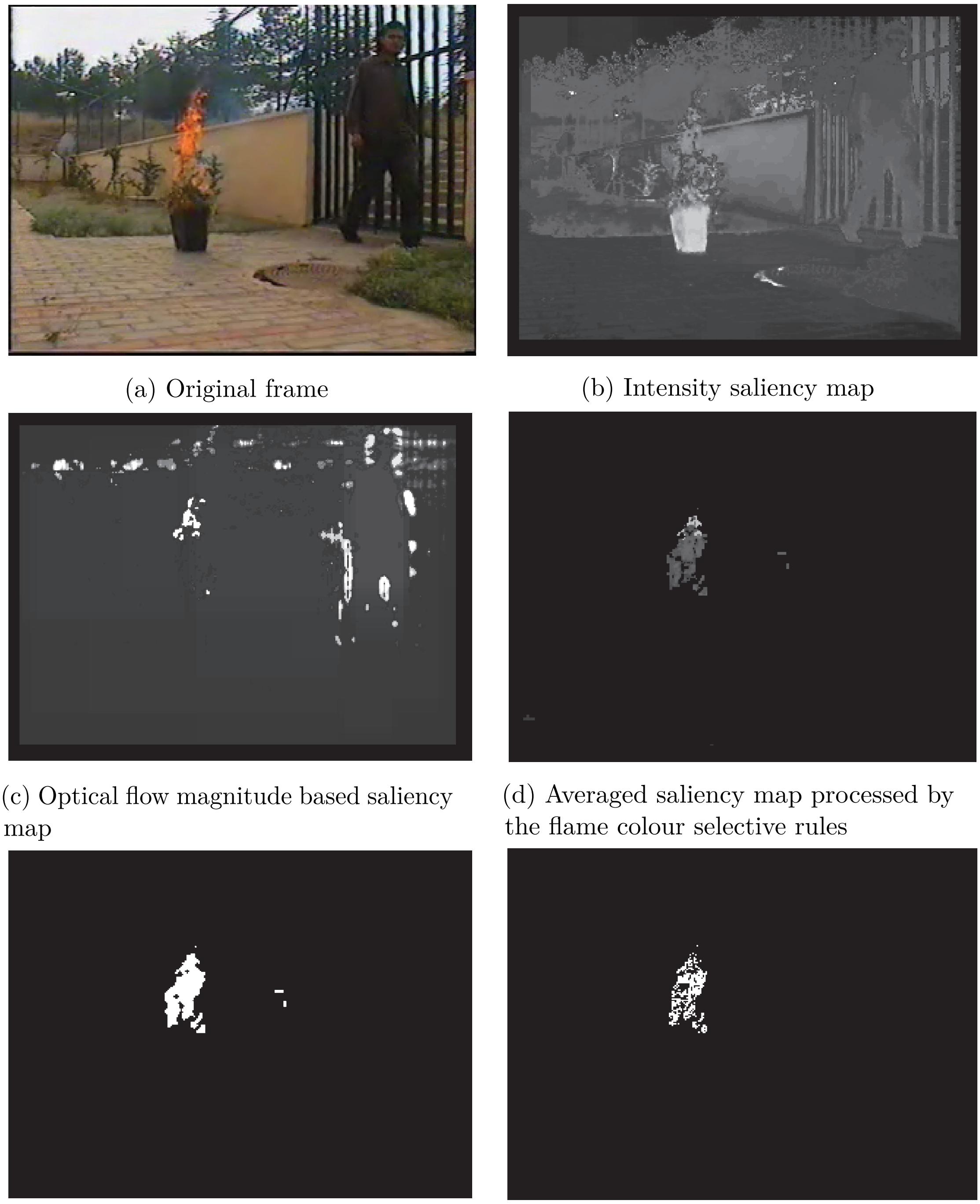
where w_F and h_F denote the width and height of each frame, respectively. Specifically, the widths and heights of the sliding window, as well as steps, are proportional to the longer side of each frame, with the ratios [0.25, 0.1, 0.01], [0.3, 0.3, 0.015], [0.5, 0.5, 0.03] and [0.7, 0.4, 0.04]. The kernels are set to be 0.9 of the window sizes.

Besides, the σs of the Gaussian blurring function in Eqs. (3.23) and (3.24) are set to 5 and 1 for the histograms of the intensity and motion saliency maps, respectively. The radius of the blurring function is set to 2σ. The results shown in Figs. 3.6 to 3.11 and Table 3.2 are obtained by the proposed framework using a combined saliency map with a threshold of 0.3. Additionally, the threshold for the number of flame pixels is set to 25 for frame-wise results, and temporal series of length 16 are used for the temporal wavelet based analysis.

3.5.2.2 Detection Results of Sample Videos

Figures 3.6 to 3.9 show the intermediate and final detection results of several sample frames from four of the testing videos processed by the proposed framework. Each figure illustrates the original frames of videos, the saliency maps based on intensity values and optical flow magnitudes, the averaged saliency maps filtered by the flame colour selective rules, the binary masks obtained by binarization and morphological operation, and the final detected regions of flames, respectively.

Among them, Figure 3.6 and Figure 3.7 are successful examples of detecting flames. It can be seen that the optical flow based saliency maps successfully highlight the regions of large motion by assigning high saliency values to them, which contributes



(a) Original frame

(b) Intensity saliency map

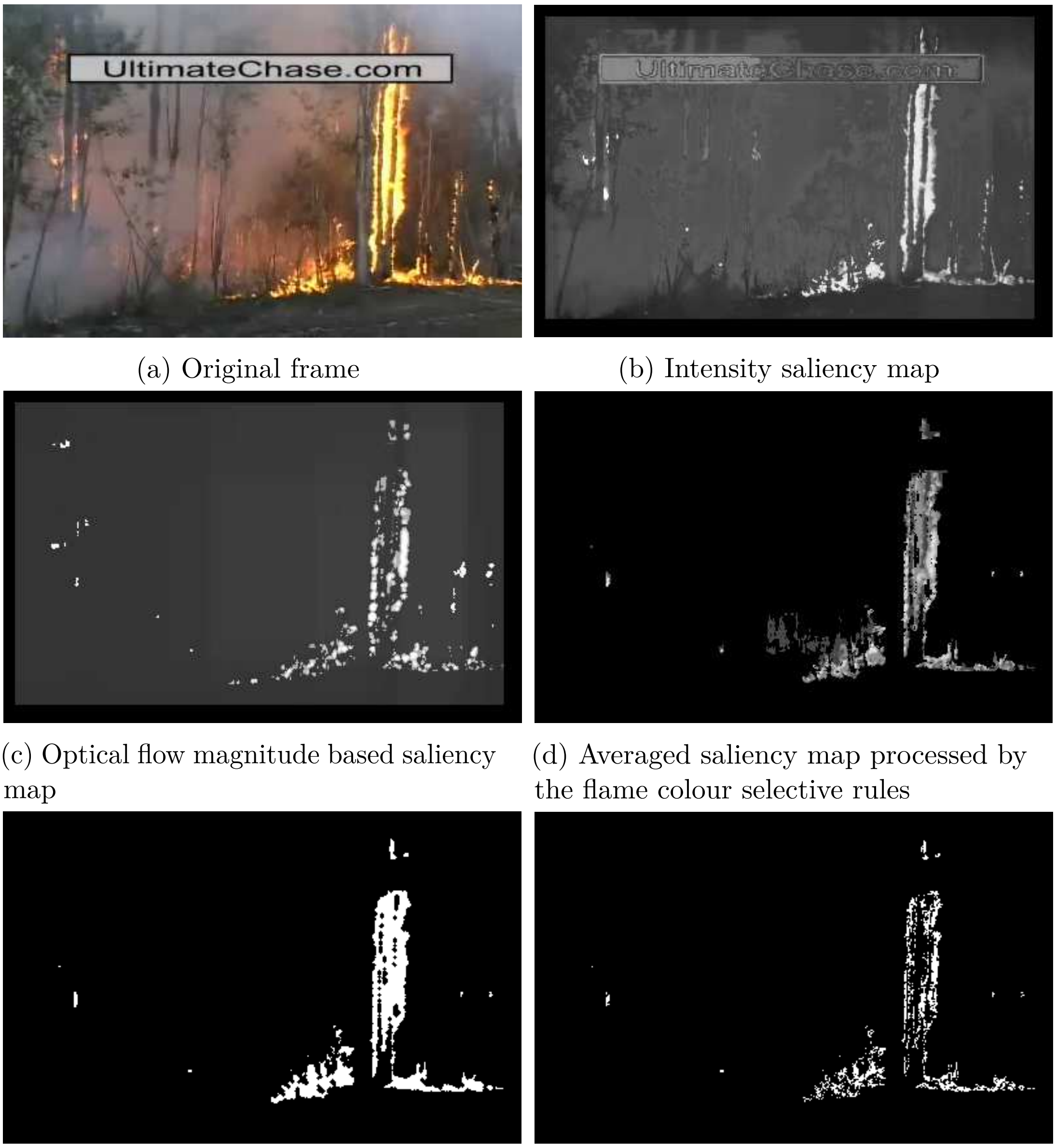
(c) Optical flow magnitude based saliency map

(d) Averaged saliency map processed by the flame colour selective rules

(e) Binary result after binarization and morphological operation

(f) Final detected flames after temporal wavelet transform based analysis

Figure 3.6. True positive detection results of the framework based on optical flow and saliency analysis tested on Video VC7, in which a man walking around burning branches.



(a) Original frame

(b) Intensity saliency map

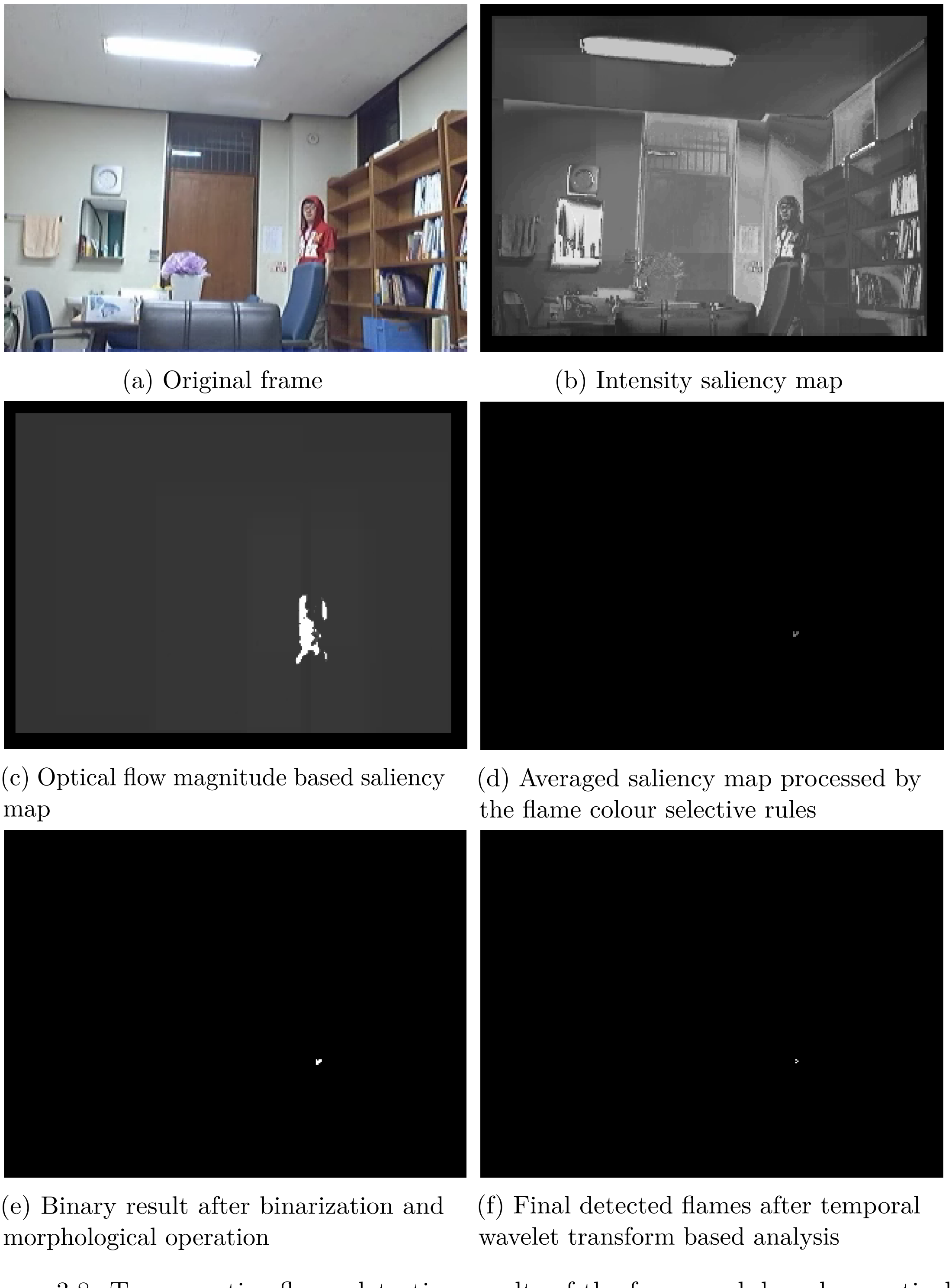
(c) Optical flow magnitude based saliency map

(d) Averaged saliency map processed by the flame colour selective rules

(e) Binary result after binarization and morphological operation

(f) Final detected flames after temporal wavelet transform based analysis

Figure 3.7. True positive detection results of the framework based on optical flow and saliency analysis tested on Video VC11, in which trees are burning and smoke exists.



(a) Original frame

(b) Intensity saliency map

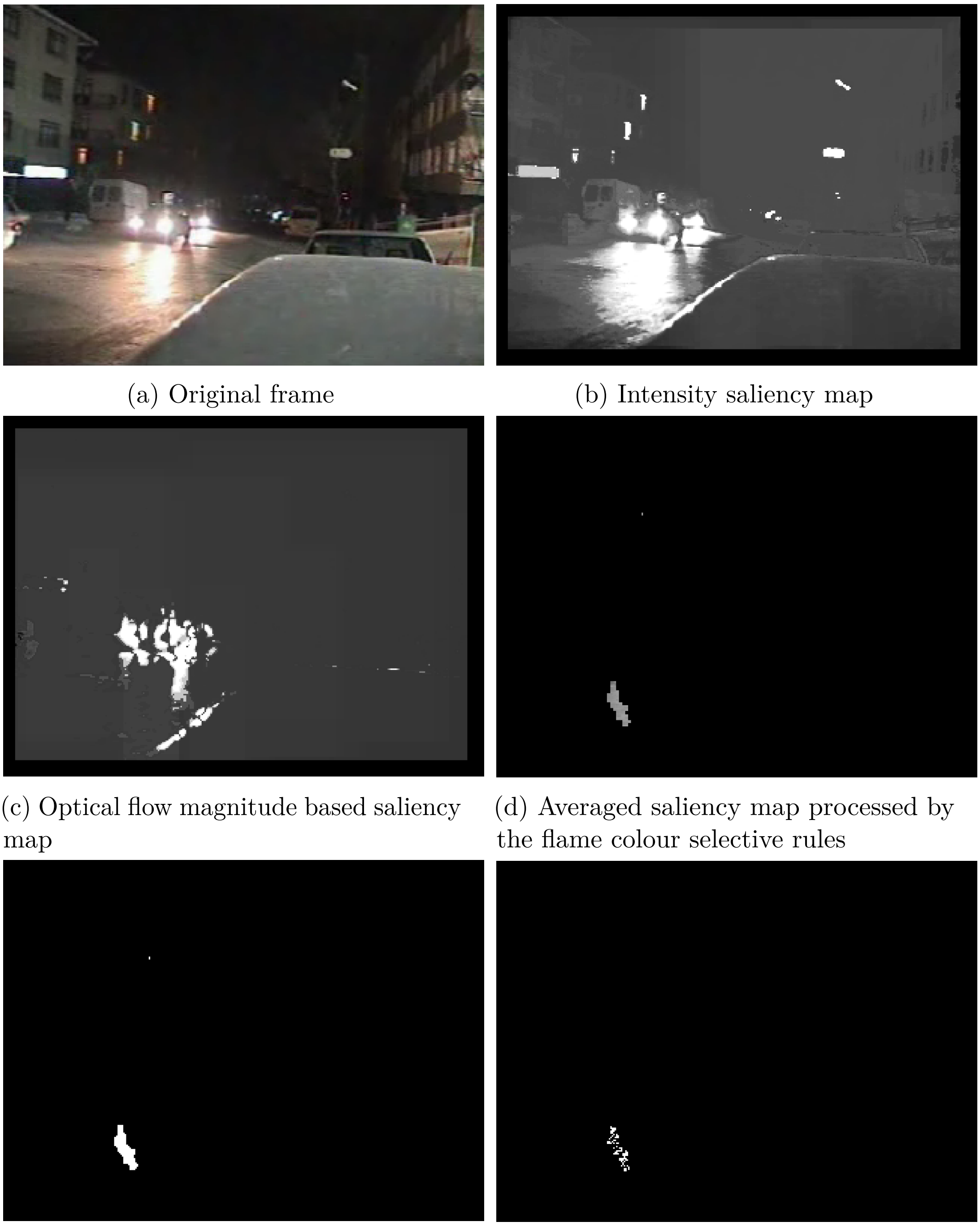
(c) Optical flow magnitude based saliency map

(d) Averaged saliency map processed by the flame colour selective rules

(e) Binary result after binarization and morphological operation

(f) Final detected flames after temporal wavelet transform based analysis

Figure 3.8. True negative flame detection results of the framework based on optical flow and saliency analysis tested on Video VC14, in which a man in red walks indoors without flames.



(a) Original frame

(b) Intensity saliency map

(c) Optical flow magnitude based saliency map

(d) Averaged saliency map processed by the flame colour selective rules

(e) Binary result after binarization and morphological operation

(f) Final detected flames after temporal wavelet transform based analysis

Figure 3.9. False positive flame detection results of the framework based on optical flow and saliency analysis tested on Video VC12, in which cars move with flashing lights without flames.

to accurate frame-wise results of the framework. Additionally, flame pixels, as well as other bright areas, obtain high values in the intensity saliency maps. The dark bucket in Figure 3.6 is also recognised as salient in the intensity saliency map because it differs from the relatively bright background nearby and the method is designed to assign high saliency values to the pixels which have different features from surroundings. Since most flames are brighter than the environments around them, the probabilistic saliency analysis approach can effectively detect flame regions based on this property. The influence of the high saliency of the bucket area in the intensity-based saliency map can be relieved by the motion-based saliency map and the flame chromatic selective rules. The final detection result shows that the proposed framework removes the interference of a walking man and effectively avoids false alarms.

From Figure 3.8, it can be seen that the saliency analysis phase together with the flame colour selective rules succeeds in discarding most of the pixels part of the walking man in red and only classifies a small number of non-flame pixels as positive. Subsequently, the temporal wavelet based analysis effectively filters out most of the falsely detected candidate flame pixels. In Figure 3.8f very few isolated pixels are detected as flames which are treated as noise by the framework and do not trigger any fire alarm for the frame.

Figure 3.9 shows an example of a false positive detection. The video was shot in a dark environment in which cars are moving with flashing headlights. Eliminating the interference caused by the car lights is challenging because the headlights move and flash in an irregular pattern which is similar to that of flames. Further research is needed to improve the performance of such videos. Features related to texture or shapes may help to distinguish flames from the objects similar to flames, such as flashing or moving lights.

3.5.2.3 Comparison with a State-of-the-art Method

Figure 3.10 and Figure 3.11 illustrate the TPRs and TNRs of the proposed framework and the method based on a deep CNN presented in [2] by Muhammad et al. In the method for comparison, a fine-tuned SqueezeNet performs classification on every frame of testing videos. It can be seen that the proposed framework in this chapter achieves TPRs higher than 90% on the majority of the testing videos containing flames except for VC7, of which a sample frame is shown in Figure 3.6. The challenges of the accurate detection on this video mainly lie in the nearly transparent colours of flames when the fire starts and nearly ceases. In contrast, the method for comparison [2] fails to detect weak or distant flames accurately, which can be seen from the TPRs below 20% of VC6, VC7, VC8, and VC10.

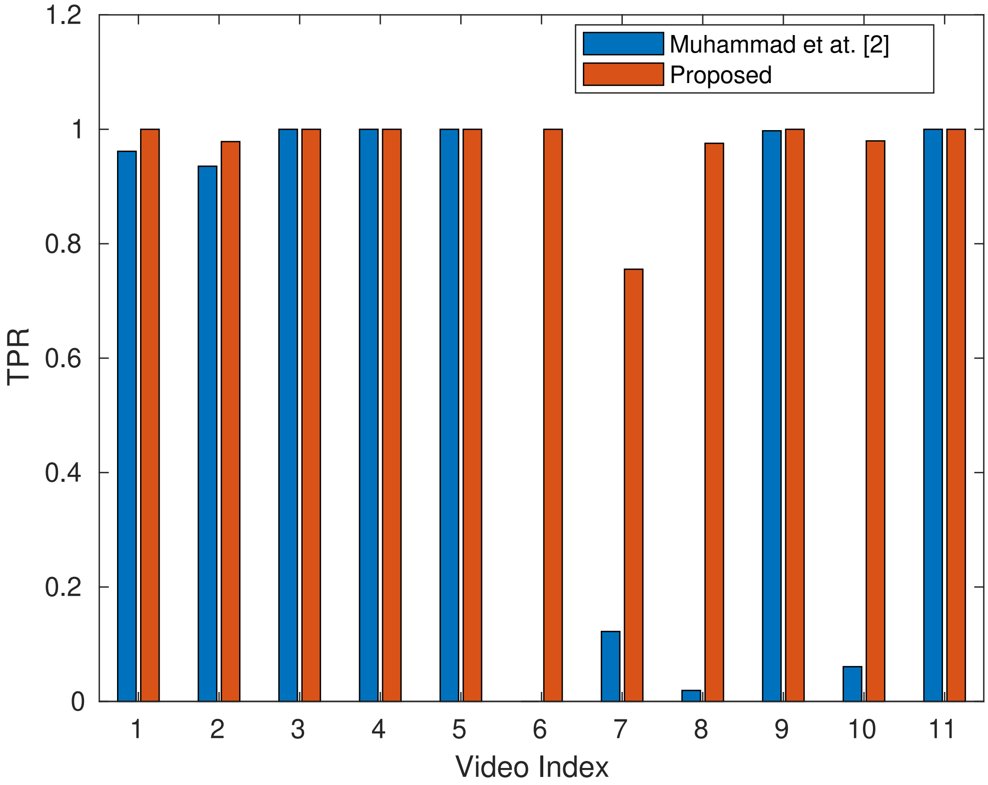


Figure 3.10. TPRs of the framework based on optical flow and saliency analysis and the method using SqueezeNet proposed by Muhammad et al. [2].

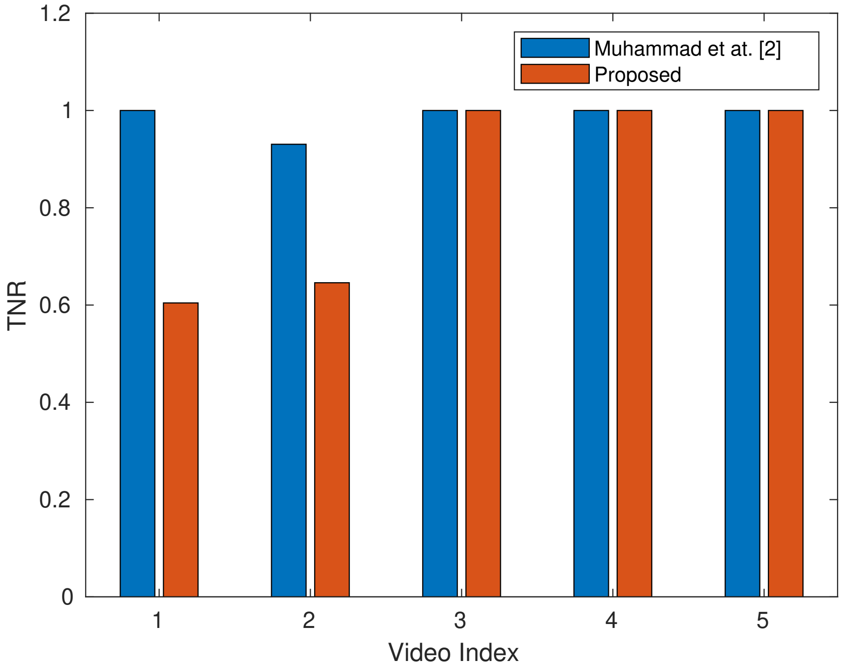


Figure 3.11. TNRs of the framework based on optical flow and saliency analysis and the method using SqueezeNet proposed by Muhammad et al. [2].

Table 3.2. Detection performance of the framework based on optical flow and saliency analysis

Videos	tp	fn	tn	fp	Total positive frames	Total negative frames	TPR	TNR
VC1	26	0	0	0	26	0	1	-
VC2	91	2	0	0	93	0	0.9785	-
VC3	48	0	0	0	48	0	1	-
VC4	41	0	0	0	41	0	1	-
VC5	214	0	0	0	214	0	1	-
VC6	176	0	0	0	176	0	1	-
VC7	519	168	5	0	687	5	0.7555	1
VC8	558	14	69	0	572	69	0.9755	1
VC9	386	0	0	0	386	0	1	-
VC10	387	8	0	0	395	0	0.9797	-
VC11	186	0	0	0	186	0	1	-
VC12	0	0	84	55	0	139	-	0.6043
VC13	0	0	93	51	0	144	-	0.6458
VC14	0	0	155	0	0	155	-	1
VC15	0	0	378	0	0	378	-	1
VC16	0	0	254	0	0	254	-	1
Overall	2632	192	1038	106	2824	1144	0.9320	0.9073

In the experiments on negative videos without flames, the proposed framework achieves TNR as high as 100% in VC14-16, showing its robustness to most interference. However, it falsely classifies some frames of VC12 and VC13 as positive, in which are moving cars with flashing headlights, as shown in Figure 3.9. The similar changing patterns of flames and flashing lights of moving cars result in several false-positive errors. In contrast, the framework for comparison tends to give negative results although the CNN is fine-tuned on roughly the same number of positive and negative images. One probable reason is that the network are trained mainly based on the features of the background if flames occupy small parts of frames.

The computational complexity of the proposed framework is influenced by the size of frames, the number of scales of sliding windows, employed saliency map (single or combined), and the threshold of saliency. Specifically, its computational burden increases with the sizes of frames and the number of scales of sliding windows for generating saliency maps. As the temporal wavelet transform based analysis

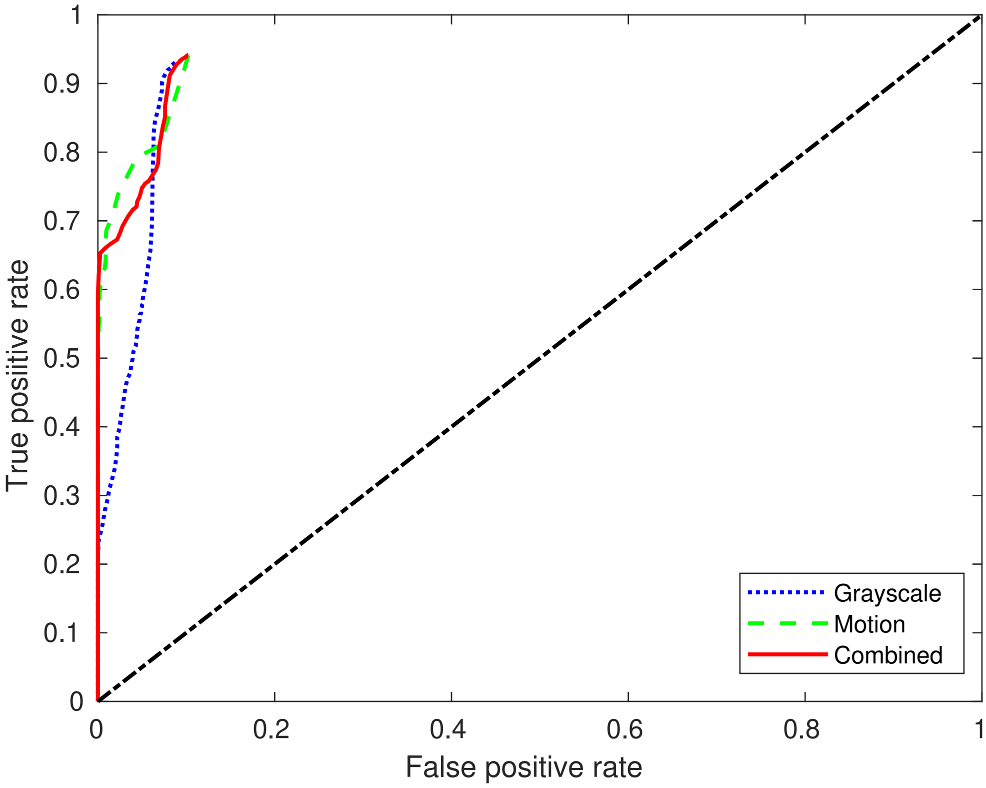


Figure 3.12. ROC curves of the proposed framework based on optical flow and saliency analysis using the intensity, motion and combined saliency maps with different thresholds of saliency.

processes all the probable flame pixels detected by the saliency map and flame colour selective rules, it needs a long time to process large candidate regions of flames, which may result from flames burning near cameras or a low saliency threshold. The average processing time of each frame differs with videos, which is in the range of 1.107s to 1.923s when using a combined saliency map with the threshold of 0.3, and four sets of sliding window sizes (as introduced in Section 3.5.2.1). The proposed framework is implemented in MATLAB and the codes are run on a computer with Intel Core i7-7700HQ CPU. The efficiency can be significantly improved by implementing the framework with the C++ language. Additionally, real-time processing can be achieved if the framework conducts detection once for every N frames instead of processing each frame of videos.

Table 3.2 illustrates the results of all testing videos of the proposed method. Generally, the proposed framework achieves an overall TPR over 93% and an overall TNR higher than 90%, which outperforms the method for comparison.

3.5.2.4 Influence of the Threshold of Saliency

The threshold of saliency will influence the final detection performance. Therefore, the results of the proposed framework with different thresholds are explored in this section, of which the ROC curve is shown in Figure 3.12. Additionally, the framework is tested with single saliency maps based on the intensity and motion, respectively, to explore the influence of these two properties of flames. The corresponding ROC curves are also illustrated in Figure 3.12. From the curves, it can be seen that frameworks with two separate and the combined saliency maps achieved similar performance when the TPRs are higher than 80%. With TPRs lower than 80%, the framework using the motion saliency map outperforms the one using the intensity-based saliency map, and the combined saliency map obtains average performance. The false positive rate (FPR) does not approach 1 because the flame colour selective rules and temporal wavelet based analysis discard some non-flame pixels. They also filter out a few flame pixels falsely, which limits the TPRs in the range of 0 to 94.12%. The flame chromic selective rules reflect the property in the brightness of flames as well, so using the motion-based saliency map together with the intensity-based one does not improve the performance much. Generally, the framework using the combined saliency map balances the dynamic and bright properties of flames, thus achieves more robust detection of flames than the frameworks with a single saliency map.

3.6 Summary

This chapter proposes a framework for autonomous flame detection, based on a probabilistic saliency analysis approach, the Horn-Schunck optical flow estimation algorithm, and a temporal wavelet transform based analysis scheme. It considers various properties of flames, namely brightness, dynamic, chromatic and flickering characteristics, which contributes to robust detection of flames. The proposed framework generates two saliency maps based on intensity values and optical flow magnitudes, respectively, and combines them for further processing. The averaged map is subsequently processed by several chromatic selective rules of flames and generates a number of potential flame pixels given a predetermined threshold. The obtained candidate flame pixels are subsequently verified by a temporal wavelet based analysis approach. A positive frame-wise result will be given if the number of detected flame pixels is larger than a threshold.

The overall TPR and TNR of the proposed framework are both higher than 90% when tested on the videos in Table 3.1, and outperforms the SqueezeNet based method for comparison. However, there are still some problems to be solved. The false alarm rates of some videos need to be improved, which may be solved by more powerful features. Additionally, novel flame chromatic models are needed to overcome the challenge of detecting weak flames.

Chapter 4

Flame Colour Model Based on Dirichlet Process Gaussian Mixture Model

Colour is an essential characteristic of flames, so it has been widely used in the tasks of flame detection. However, the colours of flames show significant diversity that results from different burning material, various intensities of combustion, and diverse illumination. Therefore, it is challenging to accurately model the colours of flames. To solve this problem, a novel flame colour model based on the DPGMM is proposed in this chapter, in which the distribution of flame colours is modelled by a GMM with a DP as its prior. The advantage of the proposed model is that the number of clusters within the mixture model can be learned from the training data without strong assumptions, which contributes to the accurate estimation of the distribution. The inference of the flame colour model is implemented by both the MCMC and VI algorithms to manage different quantities of training data. The developed DPGMM based flame colour model is incorporated into the framework introduced in Chapter 3 and improves the performance significantly.

4.1 Dirichlet Process

One of the main problems that the DP works on is about exchangeable observations which are organised into groups [76]. It is assumed that each observation $\mathbf{x}_i$ is exchangeable and sampled from a distribution with the parameter(s) $\boldsymbol{\theta}_i$ ($\mathbf{x}_i$ and $\boldsymbol{\theta}_i$ can be either scalars or vectors). The parameter $\boldsymbol{\theta}_i$ is generated from a prior

Part of the work in this chapter is published in "Zhenglin Li, Lyudmila S Mihaylova, Olga Isupova, and Lucile Rossi, "Autonomous flame detection in videos with a Dirichlet process Gaussian mixture color model," *IEEE Trans. Ind. Informat.*, vol. 14, no. 3, pp. 1146-1154, 2017".

distribution G. Thus, the model can be represented as follows:

$$\boldsymbol{\theta}_i|G \sim G \qquad \text{for each } i, \tag{4.1}$$

$$\mathbf{x}_i|\boldsymbol{\theta}_i \sim \widetilde{F}(\boldsymbol{\theta}_i) \qquad \text{for each } i, \tag{4.2}$$

where $\widetilde{F}(\boldsymbol{\theta}_i)$ is the distribution of $\mathbf{x}_i$ given $\boldsymbol{\theta}_i$. It is noteworthy that different $\boldsymbol{\theta}_i$s are exchangeable and not necessarily of distinct values. It is assumed that each parameter $\boldsymbol{\theta}_i$ is conditionally independent of others given the prior G.

Given a measurable space and a probability measure G_0 on the space [77], a DP is defined as a distribution of a probability measure G over the space, satisfying the condition that for any finite measurable partition $(A_1, ..., A_K)$ of the space, $(G(A_1), ..., G(A_K))$ is distributed according to a Dirichlet distribution with parameters of $(\alpha_0 G_0(A_1), ..., \alpha_0 G_0(A_K))$, i.e.

$$(G(A_1), ..., G(A_K)) \sim \mathrm{Dir}(\alpha_0 G_0(A_1), ..., \alpha_0 G_0(A_K)), \tag{4.3}$$

where α_0 is a positive real parameter, and $\mathrm{Dir}(\cdot)$ denotes the Dirichlet distribution. The model is denoted as $G \sim DP(\alpha_0, G_0)$ with a concentration parameter α_0 and a base distribution $G_0(\cdot\,;\lambda)$, where λ is a hyperparameter of G_0.

4.1.1 Pólya Urn Sampling Scheme

It is difficult to employ the definition of the DP directly since it is not observable itself. To solve this problem, researchers turn to the samples of DPs, which requires a specific way of sampling from DPs. The so-called Pólya Urn or Blackwell-MacQueen sampling scheme is one of the most widely accepted methods [76].

The name of the Pólya urn scheme comes from a metaphor. Imagine there is a huge urn in which you can put an infinite number of balls. The first ball to be put into the urn is painted with the colour $\boldsymbol{\theta}_1$ which is drawn from the prior G_0. From the second ball to be dropped into the urn, one can either pick a ball from the urn randomly, paint the new ball with the same colour as the picked ball, and drop both of them back to the urn, or paint the new ball in a different colour drawn from G_0. The probability of picking an existing ball in colour $\boldsymbol{\theta}_k^*$ is proportional to the number of the balls in the same colour, while the probability of painting the new ball in a newly drawn colour is proportional to α_0. The colour of a ball corresponds to the parameter(s) $\boldsymbol{\theta}_i$ of the observation $\mathbf{x}_i$, and $\{\boldsymbol{\theta}_k^*\}_{k=1}^K$ denotes the distinct values of parameters, corresponding to the distinct colours in the metaphor. The predictive probability of $\boldsymbol{\theta}_i$ is given by

$$p(\boldsymbol{\theta}_i|\boldsymbol{\theta}_1, \cdots, \boldsymbol{\theta}_{i-1}, \alpha_0, G_0) = \sum_{k=1}^{K} \left(\frac{m_k}{i-1+\alpha_0} \delta_{\boldsymbol{\theta}_k^*} \right) + \frac{\alpha_0}{i-1+\alpha_0} G_0(\boldsymbol{\theta}_i), \tag{4.4}$$

where K is the number of distinct values of parameters, $\delta_{\boldsymbol{\theta}_k^*}$ denotes the Dirac measure centred at $\boldsymbol{\theta}_{\boldsymbol{k}}^*$ [77], and $m_k = \sum_{j=1}^{i-1} \mathbb{I}(\boldsymbol{\theta}_j = \boldsymbol{\theta}_k^*)$. The indicator function $\mathbb{I}(\cdot)$ is defined as

$$\mathbb{I}(e) = \begin{cases} 1 & \text{if } e \text{ is true} \\ 0 & \text{if } e \text{ is false} \end{cases}. \tag{4.5}$$

It is important to note that both K and $\{m_k\}_{k=1}^K$ can increase with samples. Additionally, a crucial assumption is that the samples drawn from G_0 are different with probability 1.

As $\boldsymbol{\theta}_i$ is usually a vector, it is more efficient to work with the scalar variable z_i which specifies the index of the cluster (distinct parameters) associated with $\mathbf{x}_i$, i.e. $\boldsymbol{\theta}_i = \boldsymbol{\theta}_{z_i}^*$. This induces the so-called Chinese restaurant process (CRP) which will be introduced in Section 4.1.2.

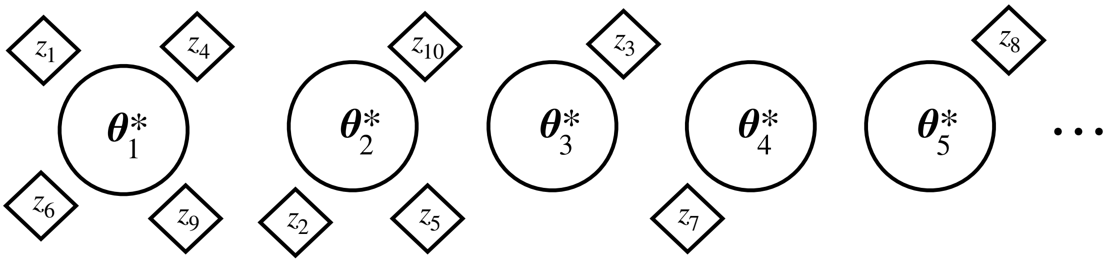
4.1.2 Chinese Restaurant Process

Imagine a boundless Chinese restaurant with an infinite number of tables in it and each table can serve unlimited customers. A sequence of customers come into the restaurant successively and randomly choose a table to sit at. The first customer is assumed to sit at the first table, while the i-th customer can either sit at an existing table together with someone else or chooses a new table with nobody around it, with the probabilities given by

$$p(z_i|z_1, ..., z_{i-1}, \alpha_0) = \sum_{k=1}^{K} \left(\frac{m_k}{i-1+\alpha_0} \mathbb{I}(z_i = k) \right) + \frac{\alpha_0}{i-1+\alpha_0} \mathbb{I}(z_i = k^*), \tag{4.6}$$

where z_i is an indicator variable specifying the table, at which the i-th customer sits, m_k is the number of customers already at table k, and the situation that the i-th customer sits at a new table is represented by $z_i = k^*$.

A schematic diagram of the CRP is provided in Figure 4.1. After all the customers have taken their seats, a partition plan of the seats of those customers (discrete variables $z_1, z_2, ...$) is obtained, contributing to a naturally clustering property of the



z_1
z_4
z_{10}
z_3
z_8
$\boldsymbol{\theta}_1^*$
$\boldsymbol{\theta}_2^*$
$\boldsymbol{\theta}_3^*$
$\boldsymbol{\theta}_4^*$
$\boldsymbol{\theta}_5^*$
...
z_6
z_9
z_2
z_5
z_7

Figure 4.1. Schematic diagram of the Chinese restaurant process

CRP. When related with the Pólya urn scheme, the customers at the k-th table share the same dish $\boldsymbol{\theta}_k^*$ (distinct parameters) drawn from the base distribution G_0, which corresponds to the distinct colours of balls. The cluster number, referred as the table number in the metaphor, is influenced by the concentration parameter α_0, as it determines the probability a sample belongs to a new cluster (how likely a customer chooses a new table rather than an existing one in the restaurant). Additionally, $z_1, z_2, ...$ are exchangeable since $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, ...$ are exchangeable. Therefore, each customer can be treated as the last one. The CRP is a distribution over a partition of integers, which is another perspective of the DP [78].

4.1.3 Stick-breaking Process

The stick-breaking process provides a way to construct a DP defined by

$$\beta_k \sim \text{Beta}(1, \alpha_0), \tag{4.7}$$

$$\boldsymbol{\theta}_k^* \sim G_0, \tag{4.8}$$

$$\pi_k = \beta_k \prod_{l=1}^{k-1} (1 - \beta_l) = \beta_k \left(1 - \sum_{l=1}^{k-1} \pi_l\right), \tag{4.9}$$

$$G = \sum_{k=1}^{\infty} \pi_k \delta_{\boldsymbol{\theta}_k^*}, \tag{4.10}$$

where Beta$(\cdot)$ denotes the Beta distribution, both $\{\beta\}_{k=1}^{\infty}$ and $\{\boldsymbol{\theta}^*\}_{k=1}^{\infty}$ are independent and identically distributed (i.i.d.) random variables, and α_0 and G_0 are the concentration parameter and base distribution of the constructed DP, respectively.

The construction definition in (4.7)-(4.10) can be interpreted metaphorically as successively breaking a stick of unit length into infinite pieces. At each step, the size to be cut from the stick is proportional to the rest part with a scale drawn independently from the distribution Beta$(1, \alpha_0)$. The distribution over $\boldsymbol{\pi} = \{\pi_k\}_{k=1}^{\infty}$ can also be expressed as $\boldsymbol{\pi} \sim \text{GEM}(\alpha_0)$, which comes from the initials of Griffiths, Engen and McCloskey [76]. From the definition in (4.7)-(4.10), it can be seen that samples from DP are discrete with probability 1.

4.1.4 Infinite Mixture Model

When the prior of (4.1) is set to a DP, which is represented by the stick-breaking process defined in (4.7)-(4.10), the model can be interpreted as a mixture model of an infinite number of clusters, which can be rewritten as

$$\boldsymbol{\pi} \sim \mathrm{GEM}(\alpha_0), \tag{4.11}$$

$$\boldsymbol{\theta}_k^* \sim G_0, \tag{4.12}$$

$$z_i \sim \boldsymbol{\pi}, \tag{4.13}$$

$$\mathbf{x}_i | z_i, (\boldsymbol{\theta}_k^*)_{k=1}^{\infty} \sim \widetilde{F}(\boldsymbol{\theta}_{z_i}^*), \tag{4.14}$$

where $\boldsymbol{\pi}$ denotes the infinite sequence of mixture weights defined in Eq. (4.9), $\{\boldsymbol{\theta}_k^*\}_{k=1}^{\infty}$ are distinct parameters drawn from the base distribution G_0, observation $\mathbf{x}_i$ is generated from the distribution $\widetilde{F}(\boldsymbol{\theta}_{z_i}^*)$, of which z_i is the indicator variable distributed according to $\boldsymbol{\pi}$. It is noteworthy that the cluster labels are not interchangeable in this mixture model, since changing labels of clusters will change the joint probability $p(\mathbf{x}_i, z_i, \boldsymbol{\pi}, \{\boldsymbol{\theta}_k^*\}_{k=1}^{\infty})$. It is because the mixture weights $\boldsymbol{\pi}$ are usually in descending order resulted from the definition in Eq. (4.9).

The mixture model does not have a fixed number of clusters. Instead, the number of clusters may increase when more data are available. In practical applications, the cluster number will not go infinitely with limited observations. Thus the infinite mixture model will have a finite number of clusters, which are learned from data based on the Bayesian inference rather than set empirically.

4.2 Flame Colour Modelling and Inference

4.2.1 Model Setting

The colours of flames are significantly diverse because of the various burning material and combustion intensities. The diversity makes it difficult to model the flame colours with empirical rules. Alternatively, modelling the distribution of flame colours based on observations can provide more accurate estimations. Theoretically, a GMM can approximate any distribution accurately by setting a proper number of components and adjusting other parameters, so it has been widely used for estimating the distributions of different variables in data-driven methods. The colours of flames can also be approximated by a GMM, of which the main difficulty lies in the determination of the number of mixture components. An improperly set number of clusters will result in an inaccurate estimation of the distribution. To solve this problem, a GMM with a DP as its prior is proposed to model the distribution of flame colours inspired by the infinite mixture model introduced in Section 4.1.4. It allows the model to learn the cluster number from training data, and thus the potential bias caused by improperly set parameters can be avoided.

As the training data are extracted from images of various illumination, the trained DPGMM based flame colour model is robust to different lighting conditions. Therefore, the colour space in which the model is established has minor influence on the estimated distribution, and thus the RGB space is taken as an example here. Denote the colour of a flame pixel as a vector $\mathbf{x}_i = [R_i, G_i, B_i]^\mathsf{T}$. Then

$$p(\mathbf{x}_i|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_i; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k), \tag{4.15}$$

where $\mathcal{N}(\cdot)$ is the Gaussian distribution, and $\boldsymbol{\mu}_k$, $\boldsymbol{\Sigma}_k$ and π_k denote the mean vector, covariance matrix, and the mixture weight of the k-th Gaussian component, respectively. Then the GMM is characterized by the parameters $\boldsymbol{\mu} = \{\boldsymbol{\mu}_1, ..., \boldsymbol{\mu}_K\}$, $\boldsymbol{\Sigma} = \{\boldsymbol{\Sigma}_1, ..., \boldsymbol{\Sigma}_K\}$, and $\boldsymbol{\pi} = \{\pi_1, ..., \pi_K\}$. The parameters of the k-th component is represented by $\boldsymbol{\theta}_k^* \triangleq \{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}$.

According to the mixture model theory [78], an arbitrary observation $\mathbf{x}_i$ is generated by first specifying a cluster indexed by z_i which is distributed according to $\boldsymbol{\pi} = [\pi_1, ..., \pi_K]$. Afterwards, $\mathbf{x}_i$ is sampled from the chosen Gaussian component with the parameter $\boldsymbol{\theta}_i = \boldsymbol{\theta}_{z_i}^* \triangleq \{\boldsymbol{\mu}_{z_i}, \boldsymbol{\Sigma}_{z_i}\}$. In the conventional method, the cluster assignment variables $\{z_i\}_{i=1}^N$ of observations $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N$ and the parameters $\{\boldsymbol{\theta}_k^*\}_{k=1}^K$ are updated alternatively until convergence with a predetermined K, and the mixture weights $\boldsymbol{\pi}$ can be estimated based on the number of observations belonging to each cluster. However, the number of clusters K of the distribution of flame colours is not intuitively known. To learn K from the training data, $\boldsymbol{\theta}_i$ is assumed to be distributed according to a DP. Thus the generative model is given by

$$\boldsymbol{\theta}_i \sim DP(\alpha_0, G_0), \tag{4.16}$$

$$\mathbf{x}_i|\boldsymbol{\theta}_i \sim \mathcal{N}(\mathbf{x}_i; \boldsymbol{\theta}_i), \tag{4.17}$$

where α_0 and G_0 denote the concentration parameter and base distribution of the DP, respectively. According to the definition of the DP, the parameters $\{\boldsymbol{\theta}_i\}_{i=1}^N$ can be generated from unbounded clusters, meaning that it does not need to set K empirically before training the model. Therefore, K is not a constant and can increase with training data. In reality, K is a finite number if given a limited number of data. The probability of a testing sample $\mathbf{x}'$ being a flame pixel based on its colour can be derived as

$$\begin{aligned} p(\mathbf{x}'|\mathbf{X}) &= \int p(\mathbf{x}'|z', \boldsymbol{\pi}, \mathbf{z}, \boldsymbol{\Theta}, \mathbf{X}) p(z'|\boldsymbol{\pi}, \mathbf{z}, \boldsymbol{\Theta}, \mathbf{X}) p(\boldsymbol{\pi}, \mathbf{z}, \boldsymbol{\Theta}|\mathbf{X}) dz' d\boldsymbol{\pi} d\mathbf{z} d\boldsymbol{\Theta} \\ &= \int p(\mathbf{x}'|z', \boldsymbol{\Theta}) p(z'|\boldsymbol{\pi}) p(\boldsymbol{\pi}, \mathbf{z}, \boldsymbol{\Theta}|\mathbf{X}) dz' d\boldsymbol{\pi} d\mathbf{z} d\boldsymbol{\Theta}, \end{aligned} \tag{4.18}$$

where $\mathbf{z} = \{z_i\}_{i=1}^N, \boldsymbol{\Theta} = \{\boldsymbol{\theta}_k^*\}_{k=1}^K$, z' is the indicator variable of $\mathbf{x}'$, $p(\mathbf{x}'|z', \boldsymbol{\Theta})$ and $p(z'|\boldsymbol{\pi})$ can be calculated according to a Gaussian distribution and a multinomial distribution, respectively. The probability $p(\mathbf{x}'|\mathbf{X})$ is available if the posterior $p(\boldsymbol{\pi}, \mathbf{z}, \boldsymbol{\Theta}|\mathbf{X})$ is known. However, the posterior is not tractable as explained in Section 2.5. Therefore, it needs to be approximated rather than calculated analytically. The approximation of the posterior is implemented by both the collapsed GS [79] and Fast-VDP [53] in this chapter. Their performance are compared and discussed in Section 4.3.

4.2.2 Gibbs Sampling for Dirichlet Process Gaussian Mixture Model

To reduce the computational complexity, only the indicator variables $\mathbf{z} = \{z_i\}_{i=1}^N$ of the training data $\mathbf{X}$ are sampled by the collapsed GS algorithm. Other parameters, i.e. the mean and covariance $\{\boldsymbol{\theta}_k^*\}_{k=1}^K$ and mixture weights $\boldsymbol{\pi}$, can be estimated based on $\mathbf{z}$. Therefore, the distribution of each indicator variable conditional on other indicator variables and observations needs to be derived for the GS algorithm. The posterior is rewritten based on the Bayes' theorem as

$$\begin{aligned} & p(z_i = k|\mathbf{z}_{-i}, \mathbf{X}, \alpha_0, G_0) \\ & = p(z_i = k|\mathbf{z}_{-i}, \mathbf{x}_i, \mathbf{X}_{-i}, \alpha_0, G_0) && (4.19) \\ & = \frac{p(z_i = k, \mathbf{x}_i|\mathbf{z}_{-i}, \mathbf{X}_{-i}, \alpha_0, G_0)}{p(\mathbf{x}_i|\mathbf{z}_{-i}, \mathbf{X}_{-i}, \alpha_0, G_0)} && (4.20) \\ & = \frac{p(z_i = k|\mathbf{z}_{-i}, \mathbf{X}_{-i}, \alpha_0, G_0) \cdot p(\mathbf{x}_i|z_i = k, \mathbf{z}_{-i}, \mathbf{X}_{-i}, \alpha_0, G_0)}{\int p(z_i|\mathbf{z}_{-i}, \mathbf{X}_{-i}, \alpha_0, G_0) \cdot p(\mathbf{x}_i|z_i, \mathbf{z}_{-i}, \mathbf{X}_{-i}, \alpha_0, G_0) dz_i} && (4.21) \end{aligned}$$

where z_i is the indicator variable of $\mathbf{x}_i$, and $\mathbf{z}_{-i} = \{z_j : j \neq i\}$ denotes the set of indicator variables of all the other observations $\mathbf{X}_{-i} = \{\mathbf{x}_j : j \neq i\}$ except $\mathbf{x}_i$. To compute the conditional distribution $p(z_i = k|\mathbf{z}_{-i}, \mathbf{X}, \alpha_0, G_0)$, $\mathbf{x}_i$ is assumed to be the last observation without loss of generality according to the exchangeability of samples of the DP and iterative mechanism of the GS algorithm.

The concentration parameter α_0 only affects the allocation variables $\mathbf{z}$, and z_i is independent of $\mathbf{X}_{-i}$ and G_0 given $\mathbf{z}_{-i}$ and α_0, so (4.21) can be further simplified as

$$p(z_i = k|\mathbf{z}_{-i}, \mathbf{X}, \alpha_0, G_0) = \frac{p(z_i = k|\mathbf{z}_{-i}, \alpha_0) \cdot p(\mathbf{x}_i|z_i = k, \mathbf{z}_{-i}, \mathbf{X}_{-i}, G_0)}{\int p(z_i|\mathbf{z}_{-i}, \alpha_0) \cdot p(\mathbf{x}_i|z_i, \mathbf{z}_{-i}, \mathbf{X}_{-i}, \alpha_0, G_0)\ dz_i} . \tag{4.22}$$

The integral on the denominator in Eq. (4.22) is a constant for different k , so the posterior can be rewritten as

$$p(z_i = k|\mathbf{z}_{-i}, \mathbf{X}, \alpha_0, G_0) \propto p(z_i = k|\mathbf{z}_{-i}, \alpha_0) \cdot p(\mathbf{x}_i|z_i = k, \mathbf{z}_{-i}, \mathbf{X}_{-i}, G_0), \tag{4.23}$$

where $k \in \{1, ..., K', k^*\}$ and K' denotes the number of clusters currently taken by $\mathbf{z}_{-i}$ while $z_i = k^*$ means $\mathbf{x}_i$ belongs to a new cluster [47]. The distribution of z_i conditional on $\{z_1, ..., z_{i-1}\}$ can be induced by the CRP introduced in Section 4.1.2. Thus, the term $p(z_i = k|\mathbf{z}_{-i}, \alpha_0)$ in (4.23) can be obtained from Eq. (4.6).

For the second term on the right-hand side of (4.23), it is rewritten as follows if the i-th observation $\mathbf{x}_i$ belongs to an existing cluster,

$$\begin{aligned} p(\mathbf{x}_i|\mathbf{X}_{-i}, z_i = k, \mathbf{z}_{-i}, G_0) &= p(\mathbf{x}_i|\mathbf{X}_{k,-i}, G_0) && (4.24)\\ &= \frac{p(\mathbf{x}_i, \mathbf{X}_{k,-i}|G_0)}{p(\mathbf{X}_{k,-i}|G_0)} && (4.25)\\ &= \frac{\displaystyle\int p(\mathbf{x}_i|\boldsymbol{\theta}_k^*)\left[\prod_{j\neq i, z_j=k} p(\mathbf{x}_j|\boldsymbol{\theta}_k^*)\right] G_0(\boldsymbol{\theta}_k^*) d\boldsymbol{\theta}_k^*}{\displaystyle\int \left[\prod_{j\neq i, z_j=k} p(\mathbf{x}_j|\boldsymbol{\theta}_k^*)\right] G_0(\boldsymbol{\theta}_k^*) d\boldsymbol{\theta}_k^*}, && (4.26) \end{aligned}$$

where $\mathbf{X}_{k,-i} = \{\mathbf{x}_j : j \neq i, z_j = k\}$ denotes all the observations belonging to the k-th cluster except $\mathbf{x}_i$, and the parameters of the cluster are represented by $\boldsymbol{\theta}_k^*$.

Similarly, if the i-th observation $\mathbf{x}_i$ belongs to a new cluster, i.e. $z_i = k^*$, the second term on the right-hand side of (4.23) becomes

$$\begin{aligned} p(\mathbf{x}_i|\mathbf{X}_{-i}, z_i = k^*, \mathbf{z}_{-i}, G_0) &= p(\mathbf{x}_i|G_0) && (4.27)\\ &= \int p(\mathbf{x}_i|\boldsymbol{\theta}^*) G_0(\boldsymbol{\theta}^*) \mathrm{d}\boldsymbol{\theta}^*, && (4.28) \end{aligned}$$

where $\boldsymbol{\theta}^*$ denotes the parameters newly drawn from the base distribution G_0.

Using the GS algorithm to iteratively update the latent variables $\{\mathbf{z}_i\}_{i=1}^N$ based on the distributions described in (4.23) - (4.28), allocation plans can be obtained after convergence, based on which other parameters of the GMM can be estimated. Specifically, the mixture weights are approximated by the ratio of observations belonging to each cluster, while $\{\boldsymbol{\theta}_k^*\}_{k=1}^K$ can be approximated using the empirical means and covariance matrices of the training data assigned to each cluster. Thus, the probability of a testing observation being part of flames based on its colour can be calculated by the trained GMM.

4.2.3 Variational Inference for Dirichlet Process Gaussian Mixture Model

The inference of the flame colour model can be implemented by the GS approach as shown in Section 4.2.2. However, the computational complexity of the GS algorithm limits the quantity of training data, which usually influences the performance of models. Therefore, the target posterior is approximated by the VI algorithm in this section.

According to the idea of mean-field VI, a family of factorized distributions is proposed to approximate the target posterior. The distribution within the family that minimizes the Kullback-Leibler divergence between itself and the exact posterior is chosen as the optimal variational distribution. This turns the approximation of the posterior into a problem of optimization, which requires the labels of clusters not to be interchangeable. Therefore, the DP is represented by the stick-breaking construction scheme in the VI framework, and the GMM modelling the colours of flames can be interpreted as an infinite mixture model. Denote $\boldsymbol{\beta} = \{\beta_k\}_{k=1}^{K}$ as the set of β_ks drawn independently from a Beta distribution, as in Eq. (4.7). $\boldsymbol{\Theta} = \{\boldsymbol{\theta}_k^*\}_{k=1}^{K}$ are distinct components sampled independently from the base distribution $G_0(\boldsymbol{\theta}^*|\lambda)$, where λ is a hyperparameter. Here the number of clusters K is not a predefined constant. Instead, it can increase given more training data. Theoretically, K can be infinite, which corresponds to the infinite mixture model in Section 4.1.4, but it is a finite number in reality with limited training data. Let $\mathbf{W} = \{\boldsymbol{\beta}, \boldsymbol{\Theta}, \mathbf{z}\}$ be the collection of all the latent variables. As $\boldsymbol{\pi}$ is defined based on $\boldsymbol{\beta}$ according to Eq. (4.9), the proposed framework approximates $\boldsymbol{\beta}$ instead of $\boldsymbol{\pi}$.

The variational distribution $q(\mathbf{W}; \phi)$ is designed as

$$q(\mathbf{W}; \phi) = \prod_{k=1}^{K} [q(\beta_k; \phi_k^{\beta})\; q(\boldsymbol{\theta}_k^*; \phi_k^{\theta^*})] \prod_{i=1}^{N} q(z_i), \tag{4.29}$$

where $q(z_i)$s are categorical distributions, $\phi = \{\phi_k^{\beta}, \phi_k^{\theta^*}\}$ with ϕ_k^{β} and $\phi_k^{\theta^*}$ denoting the parameters of distributions $q(\beta_k)$ and $q(\boldsymbol{\theta}_{\boldsymbol{k}}^*)$, respectively.

Instead of using a truncated model [80], the model in the thesis assumes that all the parameters $\phi_k = \{\phi_k^{\beta}, \phi_k^{\theta^*}\}$ are tied and are equivalent to the prior for $k > T^*$ (T^* is a preset parameter and $T^* \ll K$). Specifically, for all components $k > T^*$

$$q(\beta_k) = p(\beta_k) = \mathrm{Beta}(1, \alpha_0), \tag{4.30}$$

$$q(\boldsymbol{\theta}_k^*) = p(\boldsymbol{\theta}_k^*) = G_0(\boldsymbol{\theta}_k^*; \lambda). \tag{4.31}$$

Taking the variational distribution in Eq. (4.29) into Eq. (2.58), the free energy of $q(\mathbf{W}; \phi)$ can be rewritten. The first term $\mathbb{E}_q[\log q(\mathbf{W})]$ in Eq. (2.58) is rewritten with the variational distribution in Eq. (4.29) as follows

$$\begin{aligned}
&\mathbb{E}_q[\log q(\mathbf{W})] \\
&= \mathbb{E}_q\Bigg\{ \sum_{k=1}^{K}[\log q(\beta_k;\phi_k^{\beta}) + \log q(\boldsymbol{\theta}_k^*;\phi_k^{\theta^*})] + \sum_{i=1}^{N}\log q(z_i)\Bigg\} \qquad (4.32) \\
&= \sum_{k=1}^{K}\Bigg\{\mathbb{E}_{q_{\beta_k}}[\log q(\beta_k;\phi_k^{\beta})] + \mathbb{E}_{q_{\theta_k^*}}[\log q(\boldsymbol{\theta}_k^*;\phi_k^{\theta^*})]\Bigg\} + \sum_{i=1}^{N}\mathbb{E}_q \log q(z_i), \qquad (4.33)
\end{aligned}$$

where $\mathbb{E}_q(\cdot)$ denotes the expectation with respect to q. Based on the stick-breaking scheme, the second term in Eq. (2.58) can be expressed as

$$\begin{aligned}
&\mathbb{E}_q[\log p(\mathbf{W},\mathbf{X})] \\
&= \mathbb{E}_q\Bigg\{ \log[p(\mathbf{X}|\mathbf{z},\boldsymbol{\Theta})p(\mathbf{z}|\boldsymbol{\beta})p(\boldsymbol{\beta}|\alpha)p(\boldsymbol{\Theta}|\lambda)]\Bigg\} \qquad (4.34) \\
&= \mathbb{E}_q\left\{\log\left[\prod_{i=1}^{N}p(\mathbf{x}_i|\boldsymbol{\theta}_{z_i}^*)p(z_i|\boldsymbol{\beta})\prod_{k=1}^{K}p(\beta_k|\alpha)p(\boldsymbol{\theta}_k^*|\lambda)\right]\right\} \qquad (4.35) \\
&= \sum_{i=1}^{N}\left\{\mathbb{E}_q[\log p(\mathbf{x}_i|\boldsymbol{\theta}_{z_i}^*) + \log p(z_i|\boldsymbol{\beta})]\right\} + \sum_{k=1}^{K}\left\{\mathbb{E}_{q_{\beta_k}}[\log p(\beta_k|\alpha)] + \mathbb{E}_{q_{\theta_k^*}}[\log p(\boldsymbol{\theta}_k^*|\lambda)]\right\}. \\
&\qquad (4.36)
\end{aligned}$$

Therefore, the free energy is given by

$$\begin{aligned}
\mathcal{F} &= \sum_{k=1}^{K}\left\{\mathbb{E}_{q_{\beta_k}}\left[\log q(\beta_k;\phi_k^{\beta})\right] + \mathbb{E}_{q_{\theta_k^*}}\left[\log q(\boldsymbol{\theta}_k^*;\phi_k^{\theta^*})\right]\right\} + \sum_{i=1}^{N}\mathbb{E}_q\left[\log q(z_i)\right] \\
&\quad - \left(\sum_{i=1}^{N}\left\{\mathbb{E}_q\left[\log p(\mathbf{x}_i|\boldsymbol{\theta}_{z_i}^*)+\log p(z_i|\boldsymbol{\beta})\right]\right\}+\sum_{k=1}^{K}\left\{\mathbb{E}_{q_{\beta_k}}\left[\log p(\beta_k|\alpha)\right]+\mathbb{E}_{q_{\theta_k^*}}\left[\log p(\boldsymbol{\theta}_k^*|\lambda)\right]\right\}\right) \\
&\qquad (4.37) \\
&= \sum_{k=1}^{K}\left\{\mathbb{E}_{q_{\beta_k}}\left[\log q(\beta_k;\phi_k^{\beta}) - \log p(\beta_k|\alpha)\right] + \mathbb{E}_{q_{\theta_k^*}}\left[\log q(\boldsymbol{\theta}_k^*;\phi_k^{\theta^*}) - \log p(\boldsymbol{\theta}_k^*|\lambda)\right]\right\} \\
&\quad + \sum_{i=1}^{N}\mathbb{E}_q\left[\log q(z_i) - \log p(\mathbf{x}_i|\boldsymbol{\theta}_{z_i}^*) - \log p(z_i|\boldsymbol{\beta})\right]. \qquad (4.38)
\end{aligned}$$

According to the assumptions in Eq. (4.31) and Eq. (4.31), the $q(\beta_k;\phi_k^{\beta})$ and $q(\boldsymbol{\theta}_k^*;\phi_k^{\theta^*})$ equal to the prior distributions for $k > T^*$, so the free energy can be simplified as

$$\mathcal{F} = \sum_{k=1}^{T^*} \left\{ \mathbb{E}_{q_{\beta_k}} \left[\log \frac{q(\beta_k; \phi_k^{\beta})}{p(\beta_k|\alpha)} \right] + \mathbb{E}_{q_{\theta_k^*}} \left[\log \frac{q(\boldsymbol{\theta}_k^*; \phi_k^{\theta^*})}{p(\boldsymbol{\theta}_k^*|\lambda)} \right] \right\} + \sum_{i=1}^{N} \mathbb{E}_q \left[\log \frac{q(z_i)}{p(z_i|\boldsymbol{\beta}) p(\mathbf{x}_i|\boldsymbol{\theta}_{z_i}^*)} \right]. \tag{4.39}$$

The free energy $\mathcal{F}$ is a function of T^* sets of parameters $\{\phi_k^{\beta}, \phi_k^{\theta^*}\}_{k=1}^{T^*}$ and N distributions $\{q(z_i)\}_{i=1}^{N}$. The first two terms are truncated at level T^* because no parameters need to be optimized beyond T^*. However, the variational distribution still provides $q(z_i)$ with infinite support as it allows components beyond T^* to have non-zero probabilities. Based on the settings in Eq. (4.30) and (4.31), the free energy is nested with respect to T^*, which guarantees the existence of optimal parameters when increasing T^*. Therefore, the value of T^* is adaptive during optimization and can be initialised to one.

The optimal $q(z_i)$ is [53]

$$q(z_i = k) = \frac{\exp(E_{i,k})}{\sum_{j=1}^{\infty} \exp(E_{i,j})}, \tag{4.40}$$

where $E_{i,k}$ is defined by

$$E_{i,k} = \mathbb{E}_{q_{\boldsymbol{\beta}}}[\log p(z_i = k|\boldsymbol{\beta})] + \mathbb{E}_{q_{\theta_k^*}}[\log p(\mathbf{x}_i|\boldsymbol{\theta}_k^*)]. \tag{4.41}$$

Taking the $q(z_i)$ in Eq. (4.40) into Eq. (4.39), the free energy becomes

$$\mathcal{F} = \sum_{k=1}^{T^*} \left\{ \mathbb{E}_{q_{\beta_k}} \left[\log \frac{q(\beta_k; \phi_k^{\beta})}{p(\beta_k|\alpha)} \right] + \mathbb{E}_{q_{\theta_k^*}} \left[\log \frac{q(\boldsymbol{\theta}_k^*; \phi_k^{\theta^*})}{p(\boldsymbol{\theta}_k^*|\lambda)} \right] \right\} - \sum_{i=1}^{N} \log \sum_{k=1}^{\infty} \exp(E_{i,k}). \tag{4.42}$$

The evaluation of Eq. (4.40) and Eq. (4.42) requires computing the infinite sum $\sum_{k=T^*+1}^{\infty} \exp(E_{i,k})$, which is given by

$$\sum_{k=T^*+1}^{\infty} \exp(E_{i,k}) = \frac{E_{i,T^*+1}}{1 - \exp\{\mathbb{E}_{p_\beta}[\log(1-\beta)]\}} \tag{4.43}$$

under the assumption in Eq. (4.30) and (4.31).

The previous derivations in this section work with general Dirichlet process mixture model without specifying distribution models. Specific models need to be chosen when it comes to applications. Distributions in the exponential family, which are widely used in Bayesian non-parametric modelling, are ideal choices, because of their available analytical solutions [78, 53]. Therefore, it is assumed that

$$p(\beta_k|\alpha) = \mathrm{Beta}(\alpha_1, \alpha_2), \tag{4.44}$$

$$q(\beta_k; \phi_k^{\beta}) = \mathrm{Beta}(\phi_{k,1}^{\beta}, \phi_{k,2}^{\beta}), \tag{4.45}$$

$$p(\mathbf{x}|\boldsymbol{\theta}^*) = h(\mathbf{x}) \exp\left\{(\boldsymbol{\theta}^*)^{\mathsf{T}}\mathbf{x} - a(\boldsymbol{\theta}^*)\right\}, \tag{4.46}$$

$$p(\boldsymbol{\theta}^*|\lambda) = h(\boldsymbol{\theta}^*) \exp\left\{\lambda_1\boldsymbol{\theta}^* + \lambda_2\left[-a(\boldsymbol{\theta}^*)\right] - a(\lambda)\right\}, \tag{4.47}$$

$$q(\boldsymbol{\theta}_k^*; \phi_k^{\theta^*}) = h(\boldsymbol{\theta}_k^*) \exp\left\{\phi_{k,1}^{\theta^*}\boldsymbol{\theta}_k^* + \phi_{k,2}^{\theta^*}\left[-a(\boldsymbol{\theta}_k^*)\right] - a(\phi_k^{\theta^*})\right\}, \tag{4.48}$$

where $\alpha = \{\alpha_1, \alpha_2\}$, $\lambda = \{\lambda_1, \lambda_2\}$ are hyperparameters of the prior, and $\phi_k^{\beta} = \left\{\phi_{k,1}^{\beta}, \phi_{k,2}^{\beta}\right\}$, $\phi_k^{\theta^*} = \left\{\phi_{k,1}^{\theta^*}, \phi_{k,2}^{\theta^*}\right\}$ are variational parameters to be optimized. The logarithmic normalizer $a(\cdot)$ in the definition of the exponential family ensures the integral of the distribution equal to one [81].

The probability $q(z_i = k)$ can be calculated by Eq. (4.40) together with Eq. (4.41), using

$$\mathbb{E}_{q_{\beta_k}}[\log \beta_k] = \Psi\left(\phi_{k,1}^{\beta}\right) - \Psi\left(\phi_{k,1}^{\beta} + \phi_{k,2}^{\beta}\right), \tag{4.49}$$

$$\mathbb{E}_{q_{\beta_k}}[\log(1 - \beta_k)] = \Psi\left(\phi_{k,2}^{\beta}\right) - \Psi\left(\phi_{k,1}^{\beta} + \phi_{k,2}^{\beta}\right), \tag{4.50}$$

$$\mathbb{E}_{q_{\theta_k^*}}[\log p(\mathbf{x}_i|\boldsymbol{\theta}_k^*)] = \mathbb{E}_{q_{\theta_k^*}}\left[(\boldsymbol{\theta}_k^*)^{\mathsf{T}}\mathbf{x}_i\right] - \mathbb{E}_{q_{\theta_k^*}}\left[a(\boldsymbol{\theta}_k^*)\right], \tag{4.51}$$

where $\Psi(\cdot)$ is the digamma function, defined as the logarithmic derivative of the Gamma function [82].

The variational parameters ϕ_k^{β} and $\phi_k^{\theta^*}$ are updated as

$$\phi_{k,1}^{\beta} = \alpha_1 + \sum_{i=1}^{N} q(z_i = k), \qquad \phi_{k,2}^{\beta} = \alpha_2 + \sum_{i=1}^{N}\sum_{j=k+1}^{\infty} q(z_i = j), \tag{4.52}$$

$$\phi_{k,1}^{\theta^*} = \lambda_1 + \sum_{i=1}^{N} q(z_i = k)\mathbf{x}_i, \qquad \phi_{k,2}^{\theta^*} = \lambda_2 + \sum_{i=1}^{N} q(z_i = k). \tag{4.53}$$

The variational parameters $\{\phi_k^{\beta}, \phi_k^{\theta^*}\}_{k=1}^{T^*}$ and distributions $\{q(z_i)\}_{i=1}^{N}$ are updated iteratively by Eq. (4.40), and Eq. (4.52), Eq. (4.53) until the free energy is minimized. The flame colours are modelled by a mixture model of Gaussian distributions, which is a member of the exponential family in Eq. (4.46). In the VI algorithm of DPGMM, the component labels are distinguishable (which differs from the GS algorithm for DP in Section 4.2.2). Therefore, the VI algorithm of the DPGMM reorders the cluster labels according to the approximated sizes of components after each optimization step.

The VI algorithm of the DPGMM can be accelerated using a kd-tree [83]. A kd-tree is a binary tree, where data points stored in each child node are a subset of its parent node. The Fast-VDP algorithm constraints that all data points in each data node share the same assignment of clusters. The variational parameters are updated in a similar way to that in Eqs. (4.52) and (4.53), with the value of each data point replaced by the average of all the data in each node.

With the trained colour model, each pixel in testing videos is assigned a probability which describes how likely it is part of flames based on its colour. Flame-coloured pixels will obtain high probabilities, while other regions are likely to have lower probabilities if the distribution of flame colours is estimated accurately. Given an appropriately chosen threshold, several candidate pixels can be obtained for further processing.

4.3 Experimental Results and Discussion

4.3.1 Benchmarking Database and Experimental Settings

The performance of both the introduced DPGMM based flame colour model and the detection framework incorporating the colour model is validated in this section. The proposed flame colour model is trained and tested on images from the database of [84], using both the GS and VI algorithms. The frame-wise performance of the framework using the proposed colour model is tested on videos of various scenes (datasets from [19, 73]), of which the information is provided in Table 3.1. The 16 testing videos include 3968 frames altogether which are different from the training images of the DPGMM based flame colour model.

To compare the performance of different flame colour models without the influence of thresholds, the ROC curves [75] are illustrated in Figure 4.2 to compare the pixel-wise performance of the proposed colour model trained by both the VI and GS algorithms, and other state-of-the-art colour models. Detected flame pixels based on the proposed colour model using different thresholds are also shown and compared with the results of other flame colour models in this section.

Different from the colour model itself, the framework is evaluated by the frame-wise TPR and TNR introduced in Eqs. (3.27) and (3.28). The performance is compared with a state-of-the-art approach based on a deep CNN.

4.3.2 Performance Evaluation of the DPGMM Based Flame Colour Model and Analyses

The proposed flame colour model is trained and tested on images from the dataset of [84]. The models using the GS and VI approaches are trained with different quantities of data because of their different computational efficiency. The DPGMM based flame colour model using the GS algorithm is trained with 278,879 flame pixels, which are from 10 randomly selected images from the database mentioned above, while the model using the VI algorithm is trained on $7,742,301$ flame pixels from 100 images from the same database. Both the models are tested on the same 50 images, which are randomly selected from the database and are different from the images for training.

In the model using the GS algorithm, the concentration parameter α_0 is set to a fixed value of 1 in the experiment, and the base distribution G_0 is set to the Normal-Inverse-Wishart distribution, which is the conjugate prior of the Gaussian distribution for computational convenience. The probability density function of the Normal-Inverse-Wishart distribution $NIW(\cdot)$ is given by[85]

$$p(\boldsymbol{\mu}, \boldsymbol{\Sigma}|\boldsymbol{\mu}_0, \lambda', \mathbf{A}, \nu) = \mathcal{N}\left(\boldsymbol{\mu}; \boldsymbol{\mu}_0, \frac{1}{\lambda'}\boldsymbol{\Sigma}\right) \cdot \mathcal{W}^{-1}(\boldsymbol{\Sigma}; \widetilde{\mathbf{A}}, \nu), \tag{4.54}$$

where $\boldsymbol{\mu}_0, \lambda', \widetilde{\mathbf{A}}$ and ν are parameters of the distribution, and $\mathcal{W}^{-1}(\cdot)$denotes the inverse Wishart distribution. Given the training data $\mathbf{X} = \{\mathbf{x}_1, ..., \mathbf{x}_N\}$, the hyperparameters of the prior $G_0 \sim NIW(\boldsymbol{\mu}_0, \lambda', \widetilde{\mathbf{A}}, \nu)$ are set to

$$\boldsymbol{\mu}_{\mathbf{0}} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i, \tag{4.55}$$

$$\widetilde{\mathbf{A}} = s_0 \cdot \mathbf{I}_d, \tag{4.56}$$

$$s_0 = \frac{1}{N \cdot d}\sum_{i=1}^{N}\|\mathbf{x}_i - \boldsymbol{\mu}_{\mathbf{0}}\|_2^2, \tag{4.57}$$

where d is the dimension of $\mathbf{x}_i$, and $\mathbf{I}_d$ is an identity matrix of dimension d. The degree of freedom ν is set equal to the dimension of observations, and the scale parameter λ' is set to one in the experiment. The trained GMM has 22 mixture components after discarding those with small weights (smaller than 0.001).

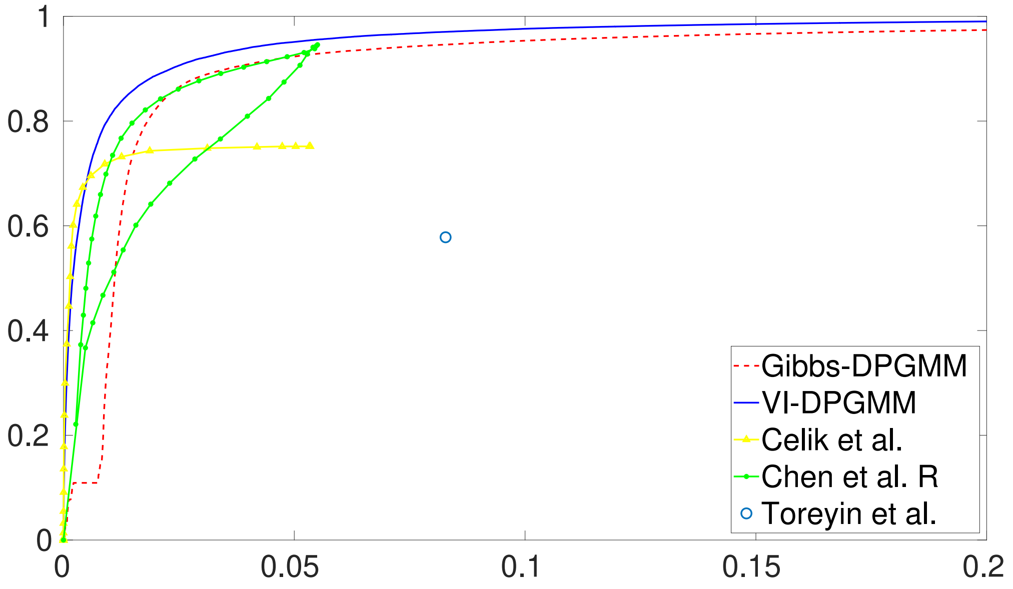
The quantities of training data for the model using the VI algorithm are much larger than the data used by the GS approach because of the high efficiency of the Fast-VDP inference algorithm which enables it to process large data in a short time. The parameter T^* in Fast-VDP is set to one at the beginning and gets the estimated cluster number of 12 after training. Both $p(\boldsymbol{\theta}_k^*|\lambda)$ and $q(\boldsymbol{\theta}_k^*; \phi_k^{\theta^*})$ are assumed to be distributed according to Gaussian-Wishart distributions [48].

The estimated cluster numbers and probability maps are not the same by the GS and VI algorithms. Two main reasons can explain the difference. One is that the variational distributions have different structures from the exact posterior, from which the GS approach samples. The other reason is that the models using the GS and VI for inference are trained with different quantities of data. Despite the difference in the estimated numbers of clusters, both trained models are able to assign high probabilities to flame-coloured pixels, which can be seen in the results shown in Figures 4.2 to 4.6 and 4.14 to 4.17.

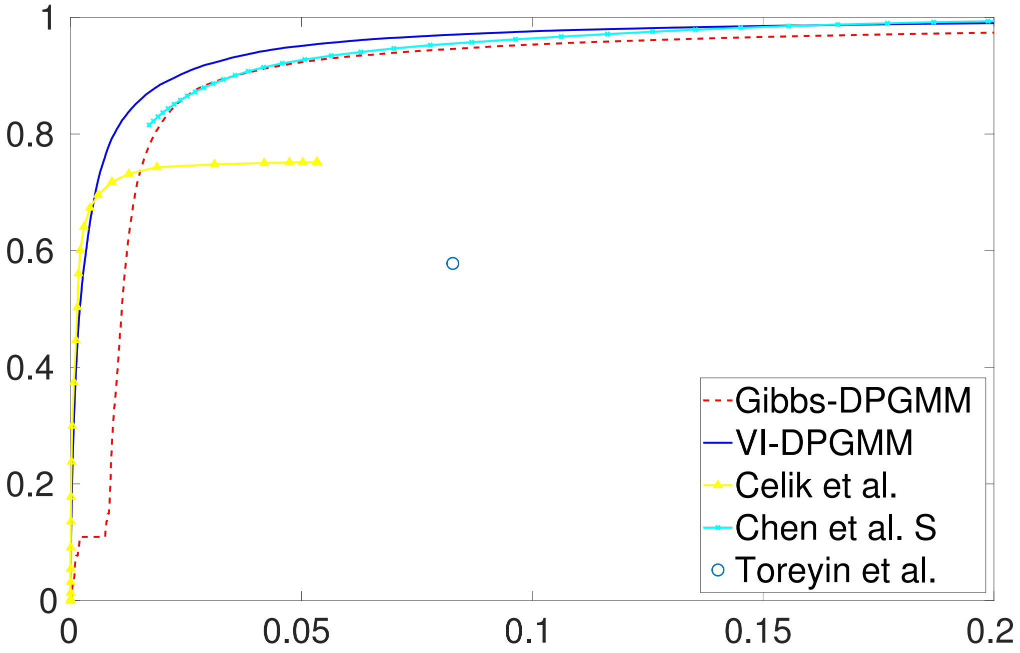
A threshold is needed in the proposed model to turn the obtained probability of each pixel into a binary result. Most existing flame colour models have thresholds in them as well [3, 4]. To compare the performance of those models, the ROC curves of the proposed model using both the GS and VI approaches are shown in Figure 4.2, together with the curves of the state-of-the-art flame colour models introduced by Chen et al. [3], Celik et al. [4], and Toreyin et al. [5, 46]. As the model of flame colours proposed by Chen et al. [3] includes two thresholds (introduced in 2.3.1), two ROC curves are obtained by changing one threshold while keeping the other fixed to the optimal value suggested in [3]. In Figure 4.2a is the curve of changing the threshold τ_R for the R channel with the threshold τ_S for the saturation set to 60, while Figure 4.2b provides the curve which changes τ_S and fixes τ_R to 125. The ROC curve of the model by Chen et al. [3] in Figure 4.2a does not change monotonically because τ_R exists in both the numerator and denominator of the inequality chromatic rules, as described in (2.14) and (2.15). The models in [3] and [4] both contain a group of inequality rules, so their FPRs do not reach 1 no matter how the thresholds are changed. Besides, the model in [5, 46] has no threshold since it conducts hard classification.

From Figure 4.2, it can be seen that the DPGMM based model using the VI algorithm outperforms all the other models. It achieves a TPR higher than 95% with the TNR smaller than 5%. The better performance over the model using the GS algorithm may result from the larger number of training data, which is consistent with the perception that a large number of training data usually contribute to more describable models.

The proposed DPGMM based flame colour model outperforms the one proposed by Toreyin et al. [5, 46], which models the distribution of flame colours by a GMM with a predetermined number of clusters. It is because the proposed model learns all the parameters, including the number of mixture components, from the training data and obtains accurate estimation of the distribution of flame colours. The model proposed by Toreyin et al. [5, 46] also assumes that the R, G and B channels are independent and each channel has the same variance for computational convenience, which is usually not true in real data. The results have proven the advantages of the proposed model experimentally. Apart from that, the model proposed by Chen et al. [3] achieves good performance on some testing images, but it is sensitive to the changes of illumination, because of the thresholds τ_R and τ_S. This limits its application to various environments. Additionally, most of the colour models conduct hard classification on pixels, while the proposed DPGMM based model estimates the

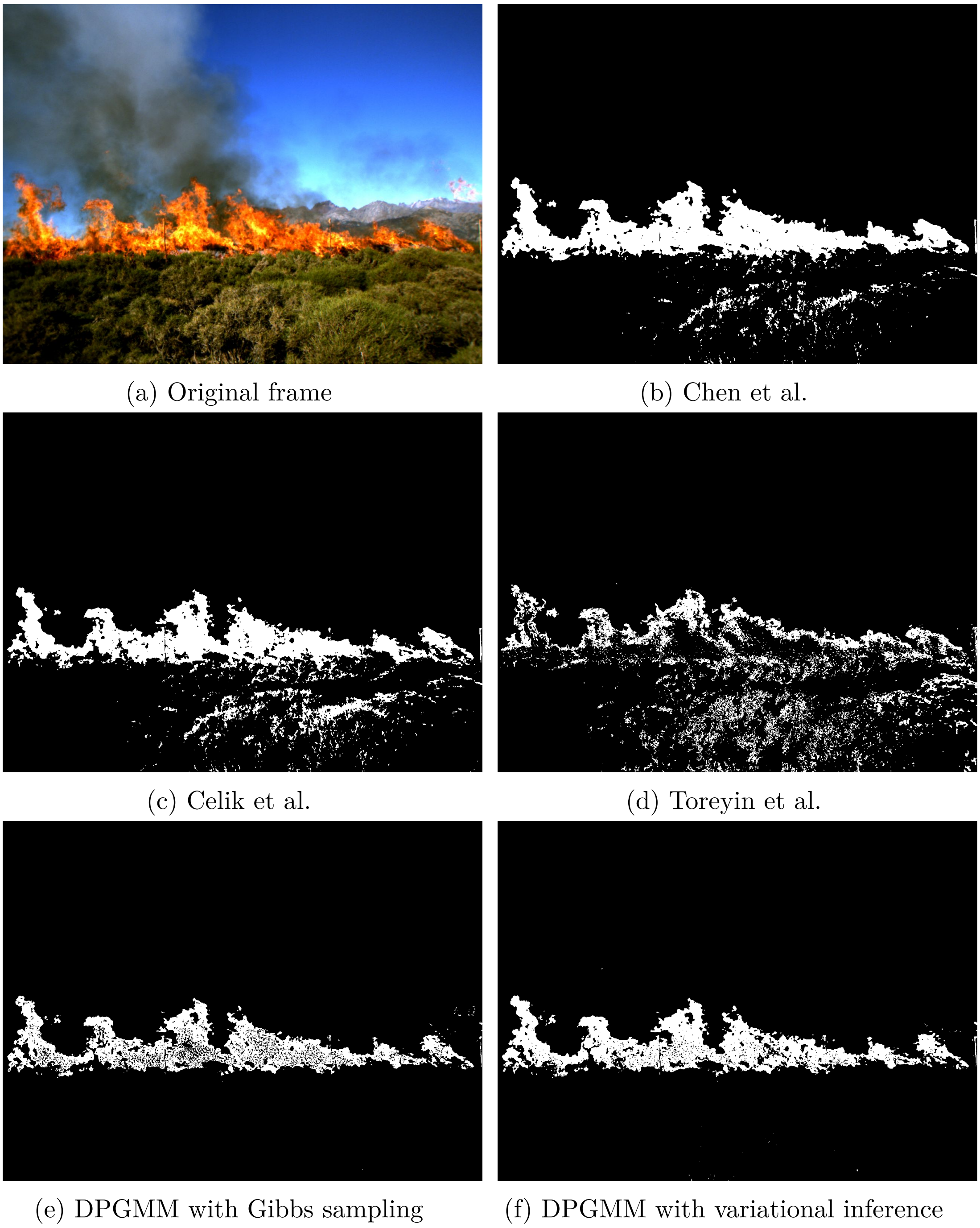


(a) ROC curves of flame colour models, among which the threshold of R is changed in the model by Chen et al. [3]



(b) ROC curves of flame colour models, among which the threshold of S is changed in the model by Chen et al. [3]

Figure 4.2. ROC curves of the proposed DPGMM based flame colour model trained by the GS and VI algorithms, and the state-of-the-art models proposed by Chen et al. [3], Celik et al. [4] and Toreyin et al. [5].



(a) Original frame

(b) Chen et al.

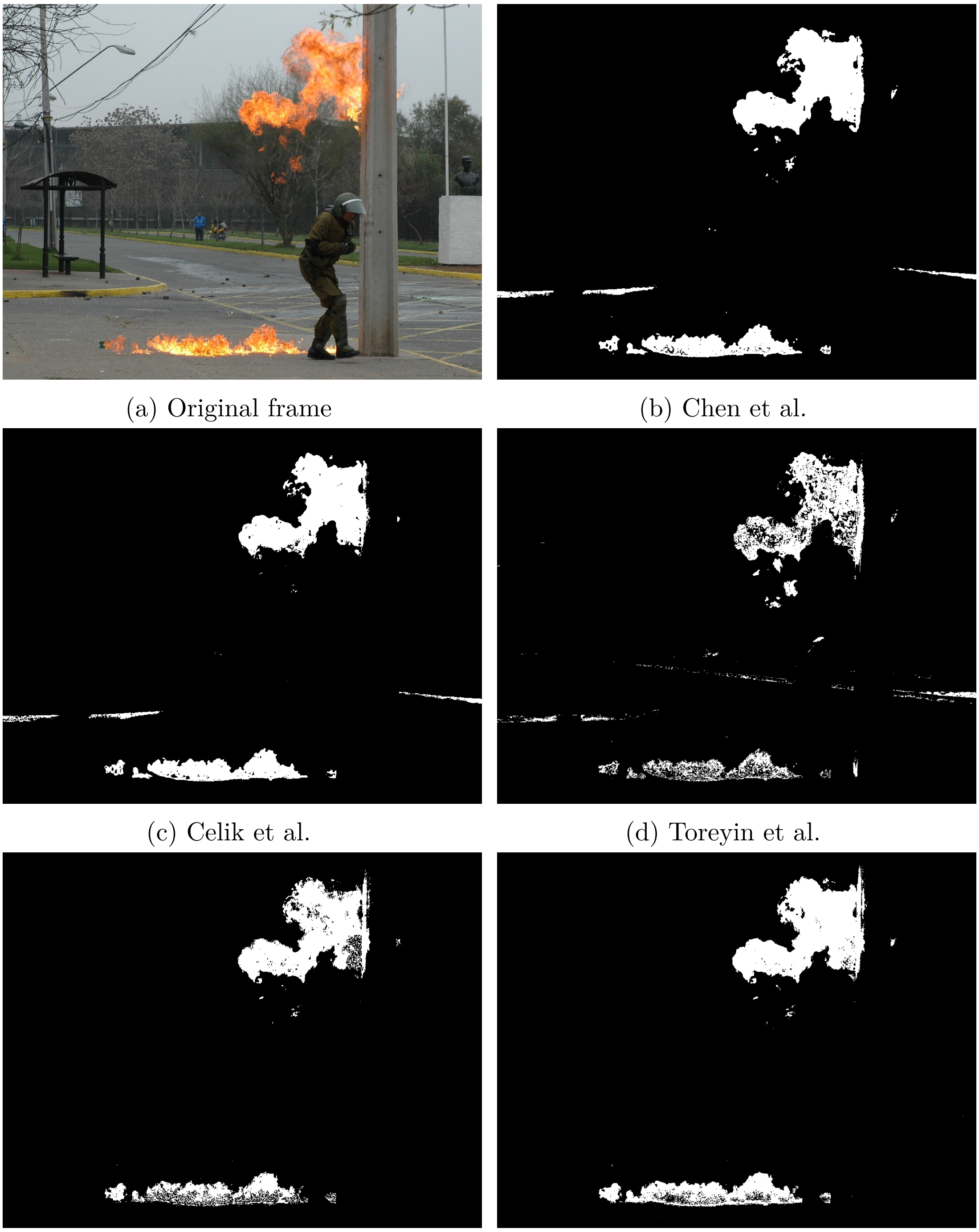
(c) Celik et al.

(d) Toreyin et al.

(e) DPGMM with Gibbs sampling

(f) DPGMM with variational inference

Figure 4.3. Detection results of the DPGMM based flame colour model and the state-of-the-art models proposed by Chen et al. [3], Celik et al. [4] and Toreyin et al. [5], tested on an image of flames in a forest fire.



(a) Original frame

(b) Chen et al.

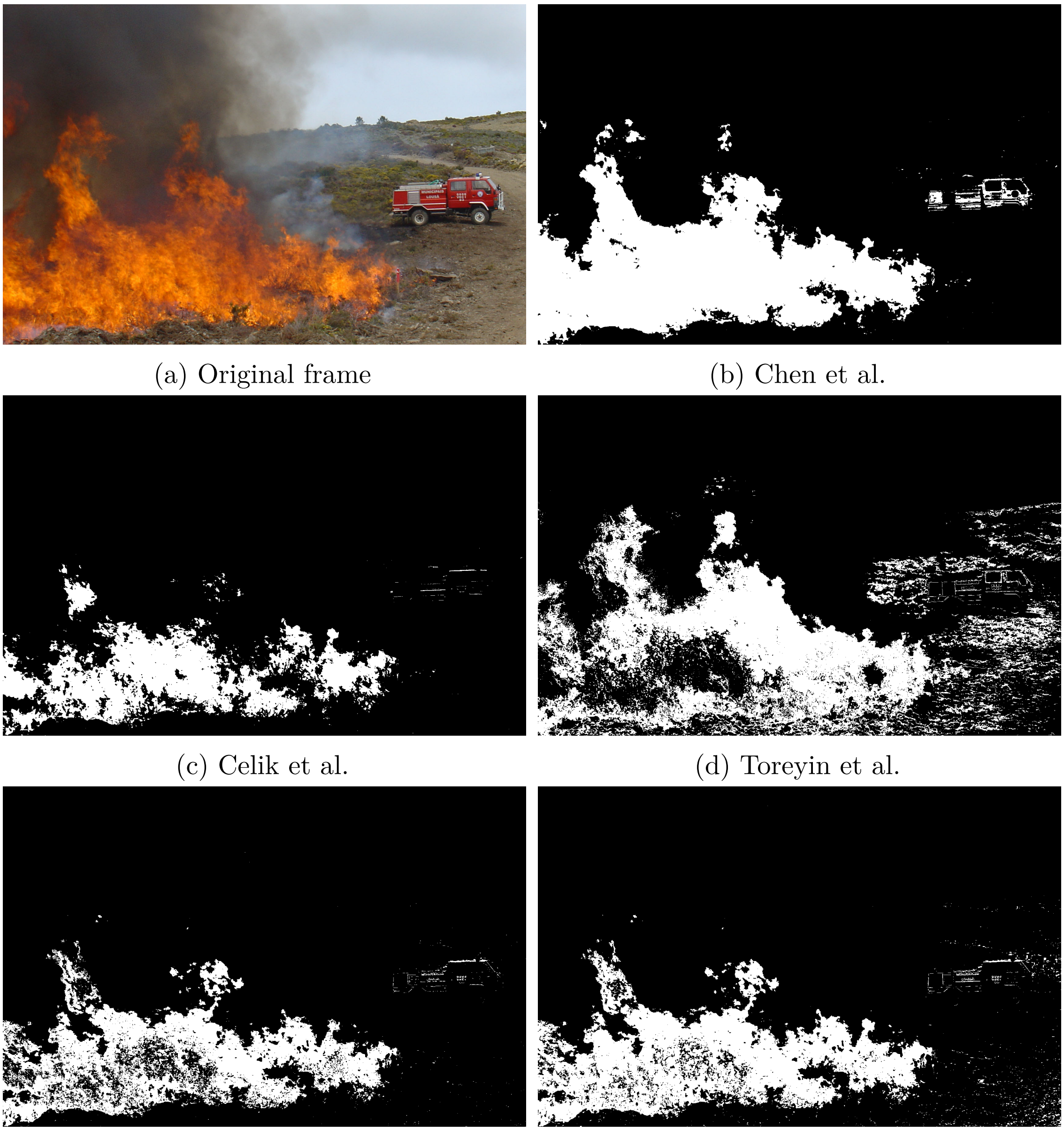
(c) Celik et al.

(d) Toreyin et al.

(e) DPGMM with Gibbs sampling

(f) DPGMM with variational inference

Figure 4.4. Detection results of the DPGMM based flame colour model and the state-of-the-art models proposed by Chen et al. [3], Celik et al. [4] and Toreyin et al. [5], tested on an image of flames in a street.



(a) Original frame

(b) Chen et al.

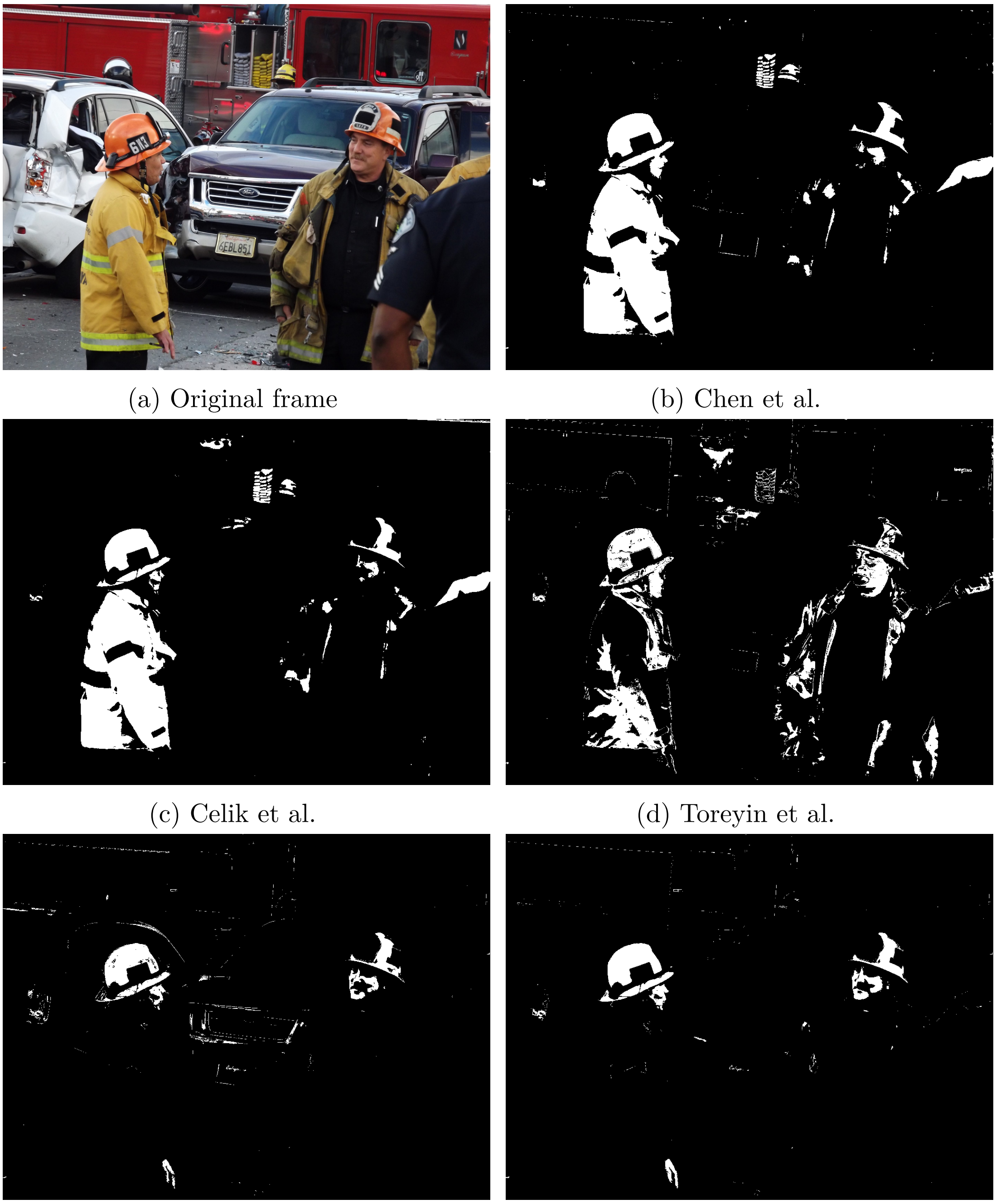
(c) Celik et al.

(d) Toreyin et al.

(e) DPGMM with Gibbs sampling

(f) DPGMM with variational inference

Figure 4.5. Detection results of the DPGMM based flame colour model and the state-of-the-art models proposed by Chen et al. [3], Celik et al. [4] and Toreyin et al. [5], tested on an image of a fire truck and flames on branches.



(a) Original frame

(b) Chen et al.

(c) Celik et al.

(d) Toreyin et al.

(e) DPGMM with Gibbs sampling

(f) DPGMM with variational inference

Figure 4.6. Detection results of the DPGMM based flame colour model and the state-of-the-art models proposed by Chen et al. [3], Celik et al. [4] and Toreyin et al. [5], tested on an image of fire trucks and fire fighters in high-visibility workwear without flames.

probabilities of pixels being flame-coloured, which can be easily combined with other steps of flame detection frameworks flexibly.

The thresholds of the models using the GS and VI algorithms are set to -14.4441 and 1.4, respectively, to produce the results shown in Figures 4.3 to 4.6. The thresholds of the proposed flame colour models are all set for logarithmic probabilities in this thesis. The thresholds τ_S and τ_R of the colour model proposed by Chen et al. [3] are set to 60 and 125, as suggested in the paper. The results in Figures 4.3 to 4.5 show that the proposed flame colour model succeeds in detecting most flame pixels, including those behind thick smoke. Additionally, it can discard the pixels of artificial objects in red to yellow colours with high saturation values. As shown in Figure 4.5, the proposed model prunes out most pixels of a flame-coloured trunk, which works better than other models. The testing image in Figure 4.6 shows a common scene in daily life which may cause false alarms of automatic flame detection system based on videos. Fewer non-flame pixels are falsely detected as flame ones by the proposed colour model than others, meaning that it can effectively reduce the false alarms caused by non-flame objects, such as vehicles or clothes in red to yellow colours. Although the proposed model using the GS algorithm fails to detect some pixels of the inner parts of flames, the influence is relieved by other steps of the framework, i.e. the median filter and temporal wavelet based analysis, and thus will not influence the final detection results.

4.3.2.1 Influence of the Number of Training Images

The proposed flame colour model based on the DPGMM can be trained with a large number of data by the VI algorithm, which contributes to an accurate estimation of the distribution of flame colours. However, a colour model trained by the VI algorithm with more observations usually requires larger memory for storing data and has higher computational complexity in predicting the probabilities of the testing pixels compared with a model trained with fewer data. Therefore, the influence of the number of training images on the performance of the flame colour model is explored in this section. Different colour models are trained by the VI algorithm with 50, 100, 200, 300, and 400 images that are randomly selected from the database [84], and are tested on the same 50 testing images. According to the ROC curves of those models shown in Figure 4.7, only small differences can be observed between the performance of different colour models. The models trained with 100, 200 and 300 images achieve slightly better results than the models trained with 50 and 400 images. It is interesting to notice that the model trained with 400 images does not outperform others, of which the reasons need to be explored. It is noteworthy that only flame pixels of the training images are used for training the colour models. To reduce the memory and computational complexity, the colour model trained with

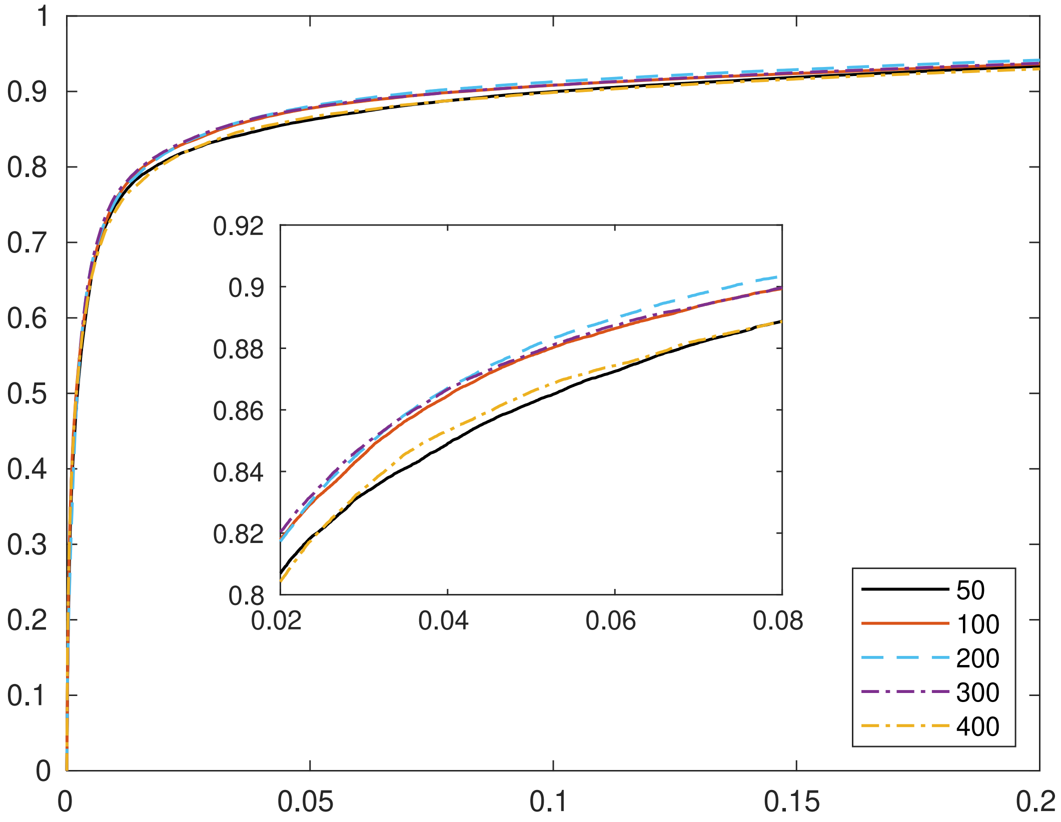


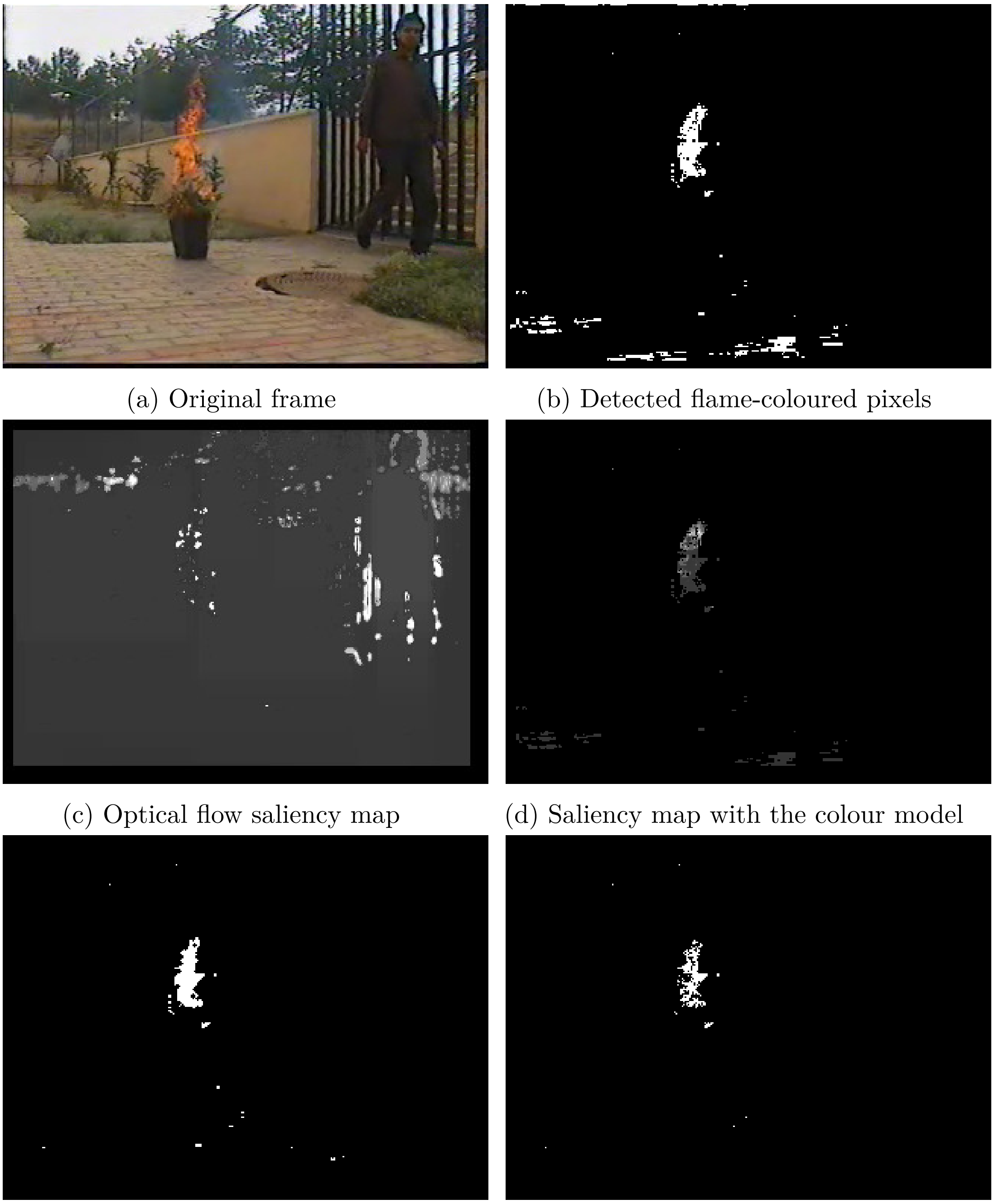
Figure 4.7. The ROCs of DPGMM based flame colour models using the VI algorithm trained with different numbers of images tested on the same data.

100 images is incorporated into the framework for detecting flames in videos in the experiments of this chapter.

4.3.3 Performance Evaluation of the Framework Using the DPGMM Based Colour Model in Videos and Discussion

The proposed DPGMM based flame colour model with the VI algorithm is incorporated into the framework introduced in Chapter 3 instead of the flame colour selective rules. A threshold of 1.9 is selected for the logarithmic probabilities obtained from the trained flame colour model. A large threshold is employed in the frame-wise detection of flames in videos to enhance the TNR of the framework. The non-flame pixels which are not discarded based on colours will be removed by other steps of the framework. Additionally, the motion saliency threshold is set to 0.21. A frame is considered as containing flames if the number of detected flame pixels is larger than 25.

Figures 4.8 to 4.10 show the results of some sample frames from testing videos, which are processed by the framework incorporating the proposed flame colour model using the VI algorithm. They show not only the final detected pixels but also intermediate results of each step of the framework.



(a) Original frame

(b) Detected flame-coloured pixels

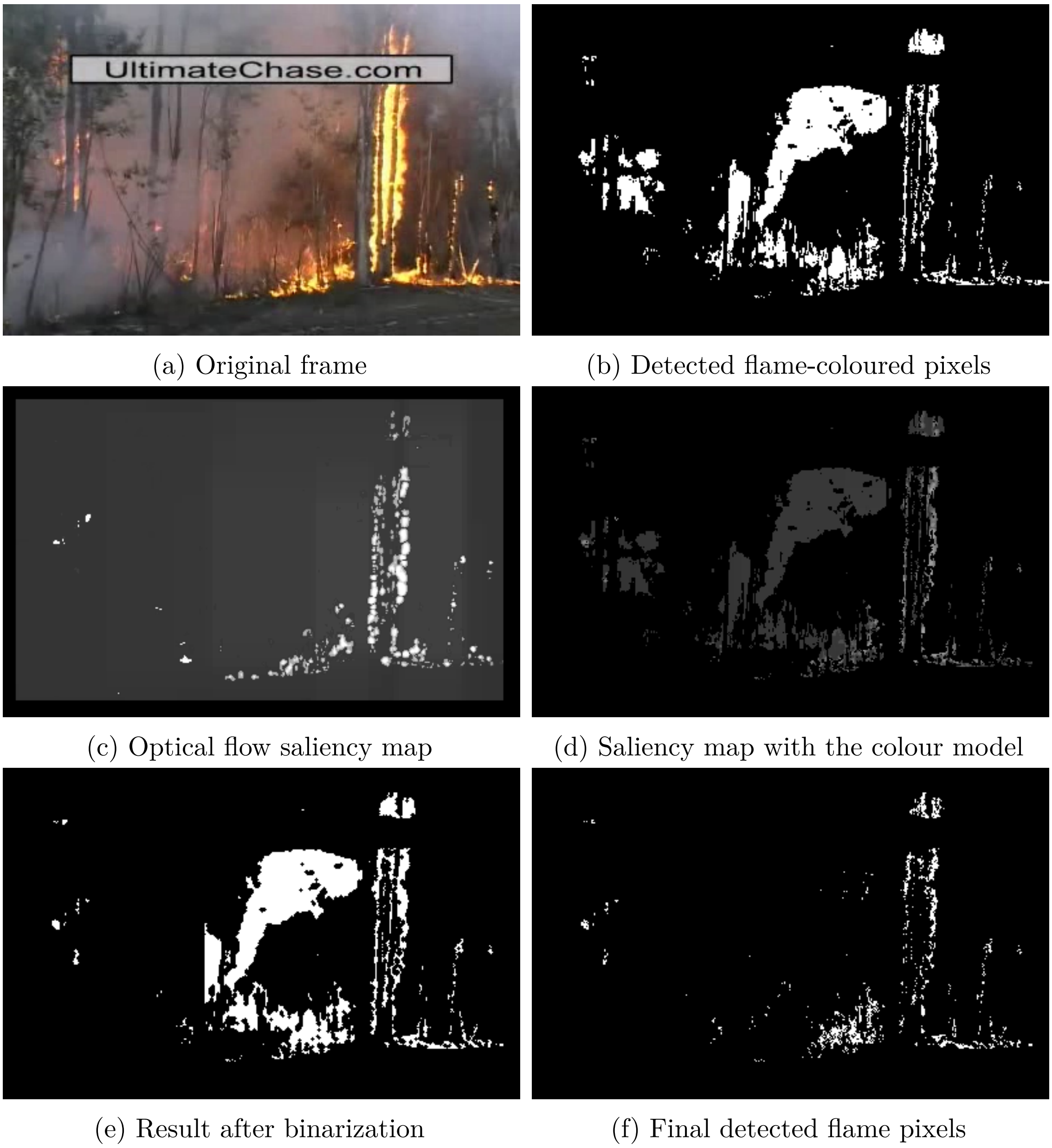
(c) Optical flow saliency map

(d) Saliency map with the colour model

(e) Result after binarization

(f) Final detected flame pixels

Figure 4.8. True positive results of the framework using the proposed DPGMM based flame colour model tested on Video VC7, in which a man walking around burning branches.



(a) Original frame

(b) Detected flame-coloured pixels

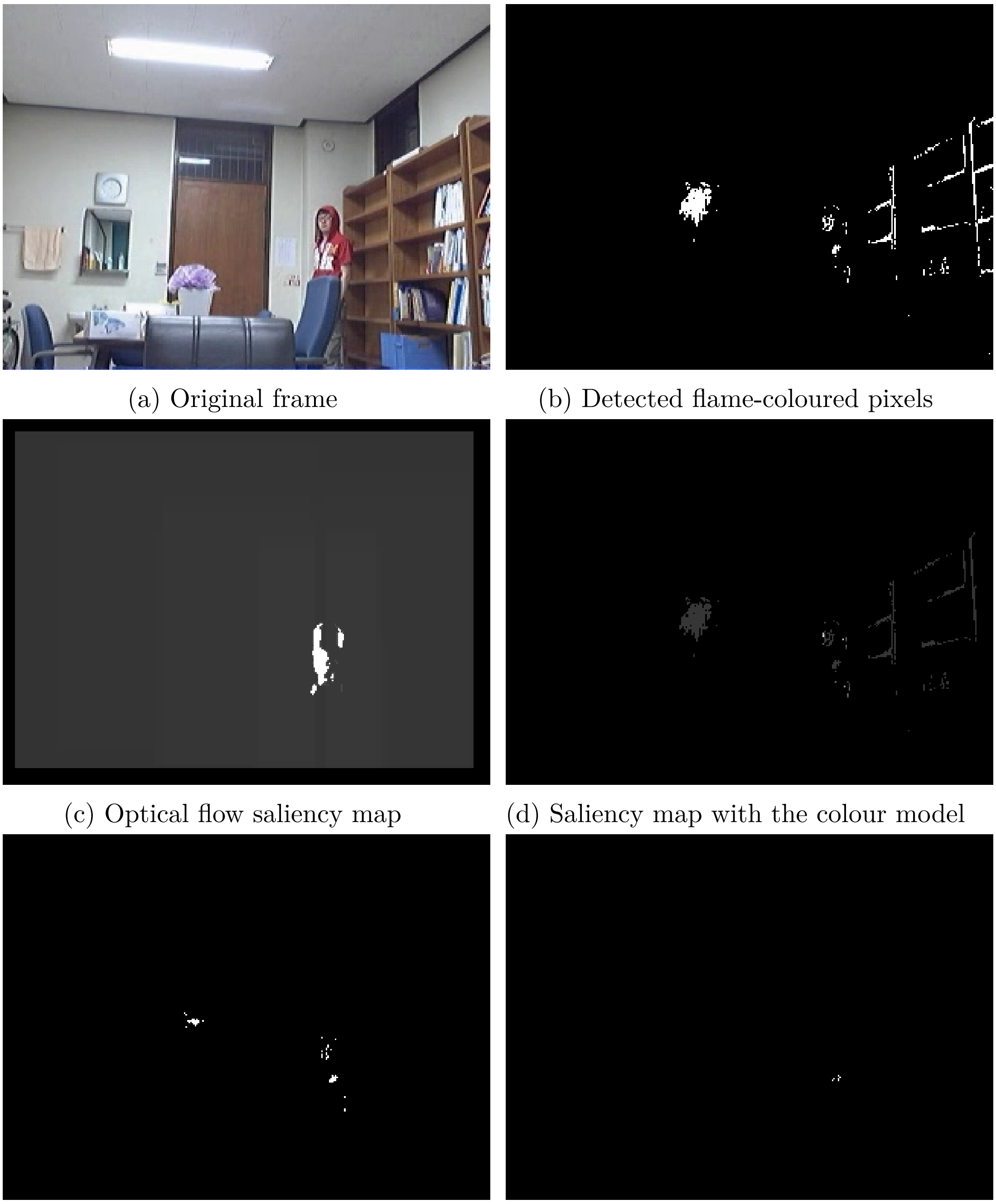
(c) Optical flow saliency map

(d) Saliency map with the colour model

(e) Result after binarization

(f) Final detected flame pixels

Figure 4.9. True positive results of the framework using the proposed DPGMM based flame colour model tested on Video VC11, in which trees are burning and smoke exists.



(a) Original frame

(b) Detected flame-coloured pixels

(c) Optical flow saliency map

(d) Saliency map with the colour model

(e) Result after binarization

(f) Final detected flame pixels

Figure 4.10. True negative results of the framework using the proposed DPGMM based flame colour model tested on Video VC14, in which a man in red walks indoors without flames.

It can be seen that the DPGMM based model detects most flame-coloured regions for further processing, which helps to enhance the TPR of the framework. Although parts of the background or smoke are detected as candidate flame pixels by the trained colour model (resulted from the reflection of lights emitted by flames), they are pruned out by the saliency map of motion and temporal wavelet transform based analysis.

Figure 4.11 and Figure 4.12 illustrate the frame-wise detection results of the proposed framework compared with the method by Muhammad et al. [2]. It can be seen that the framework using the proposed flame colour model achieves accurate detection of flames on most of the testing videos. It overwhelms the method for comparison in TPR with the overall TPR higher than 99%. It also achieves a significant improvement on TPR compared with the framework using flame colour selective rules introduced in Chapter 3, especially in the video VC7 shown in Figure 4.8. The challenge of this testing video lies in the semi-transparent colours of the weak flames at the beginning and the end of the video and the distractions caused by a walking man nearby. The proposed framework achieves better performance than the method for comparison [2] mainly because the DPGMM based colour model can detect most flame pixels for further processing. On the contrary, both the flame colour selective rules used in Chapter 3 and the trained SqueezeNet introduced in [2], fail to detect the pixels of weak flames, resulting in lower TPRs of the frameworks. Usually, weak flames at the beginning of fires are in nearly transparent colours, and thus are difficult to be detected. Better performance on these situations means earlier detection of fires which can reduce the injuries and financial loss.

Furthermore, the framework using the proposed flame colour model is robust to the interference in most testing videos, but it falsely classifies some negative frames as flame ones in videos VC12 and VC13. The challenges of these two videos mainly lie in the similar appearance and changing patterns of the flashing car lights to flames. Additionally, the false positive errors also result from the high sensitivity of the proposed framework which is designed to avoid the huge losses caused by fires. The detailed performance of the framework incorporating the DPGMM based flame colour model is shown in Table 4.1.

4.3.4 Influence of the Threshold for Probabilities of Being Flame-coloured

To explore the influence of the threshold for the estimated flame colour probabilities, the proposed model trained by the VI algorithm is tested on both videos and images with different thresholds. Figure 4.13 shows the performance and computational complexity of the proposed framework with different thresholds. In Figure 4.13a, the TPR remains stable with the threshold increasing from 0.5 to 1.5, and declines

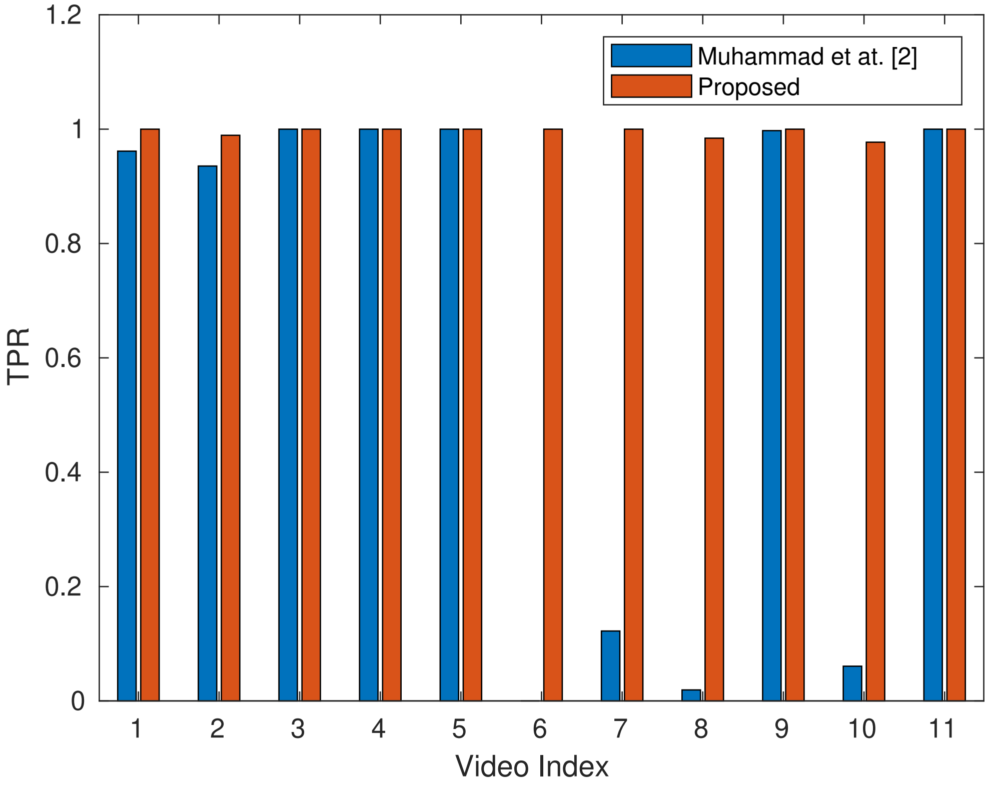


Figure 4.11. TPRs of the framework using the proposed DPGMM based flame colour model and the method using SqueezeNet proposed by Muhammad et al. [2].

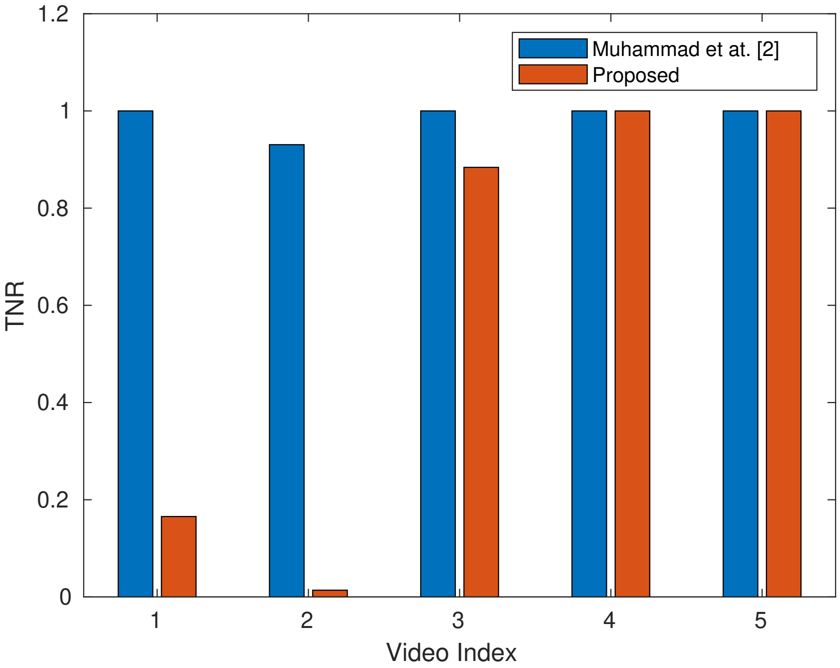


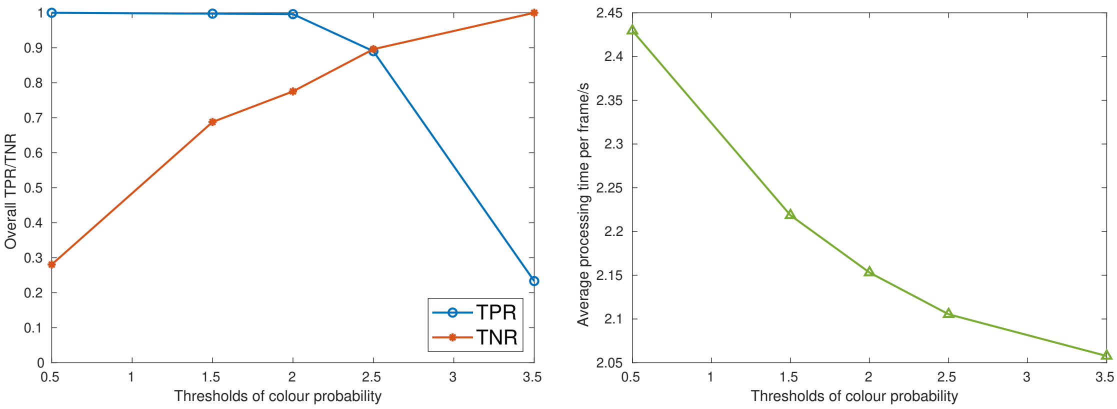
Figure 4.12. TNRs of the framework using the proposed DPGMM based flame colour model and the method using SqueezeNet proposed by Muhammad et al. [2].

Table 4.1. Detection performance of the framework using the proposed DPGMM based flame colour model

Videos	tp	fn	tn	fp	Total positive frames	Total negative frames	TPR	TNR
VC1	26	0	0	0	26	0	1	-
VC2	92	1	0	0	93	0	0.9892	-
VC3	48	0	0	0	48	0	1	-
VC4	41	0	0	0	41	0	1	-
VC5	214	0	0	0	214	0	1	-
VC6	176	0	0	0	176	0	1	-
VC7	687	0	0	5	687	5	1	0
VC8	563	9	69	0	572	69	0.9843	1
VC9	386	0	0	0	386	0	1	-
VC10	386	9	0	0	395	0	0.9772	-
VC11	186	0	0	0	186	0	1	-
VC12	0	0	23	116	0	139	-	0.1655
VC13	0	0	2	142	0	144	-	0.0139
VC14	0	0	137	18	0	155	-	0.8839
VC15	0	0	378	0	0	378	-	1
VC16	0	0	254	0	0	254	-	1
Overall	2805	19	863	281	2824	1144	0.9933	0.7544

sharply when the threshold is increased to 3.5. In contrast, the TNR increases monotonically with the threshold of the colour probability. Therefore, the threshold needs to be set properly to balance the TPR and TNR. Furthermore, a low threshold will lead to a large number of candidate flame pixels for further processing, which usually increases the computational burden. It is proven by the results shown in Figure 4.13b.

Figures 4.14 to 4.17 illustrates the estimated probabilities obtained from the trained flame colour models by both the GS and VI algorithms using colourmaps, of which the original testing images are shown in Figures 4.3 to 4.6. The detected flame-coloured pixels with different thresholds are shown as well to explore the influence of the threshold. Those binary result images are processed by a 2-D median filter with the window size of 5 × 5 to decrease the influence of noise, which is commonly used in the frameworks of automatic flame detection in videos. From the figures, it can be seen that a majority of flame pixels are assigned higher probabilities than other pixels,

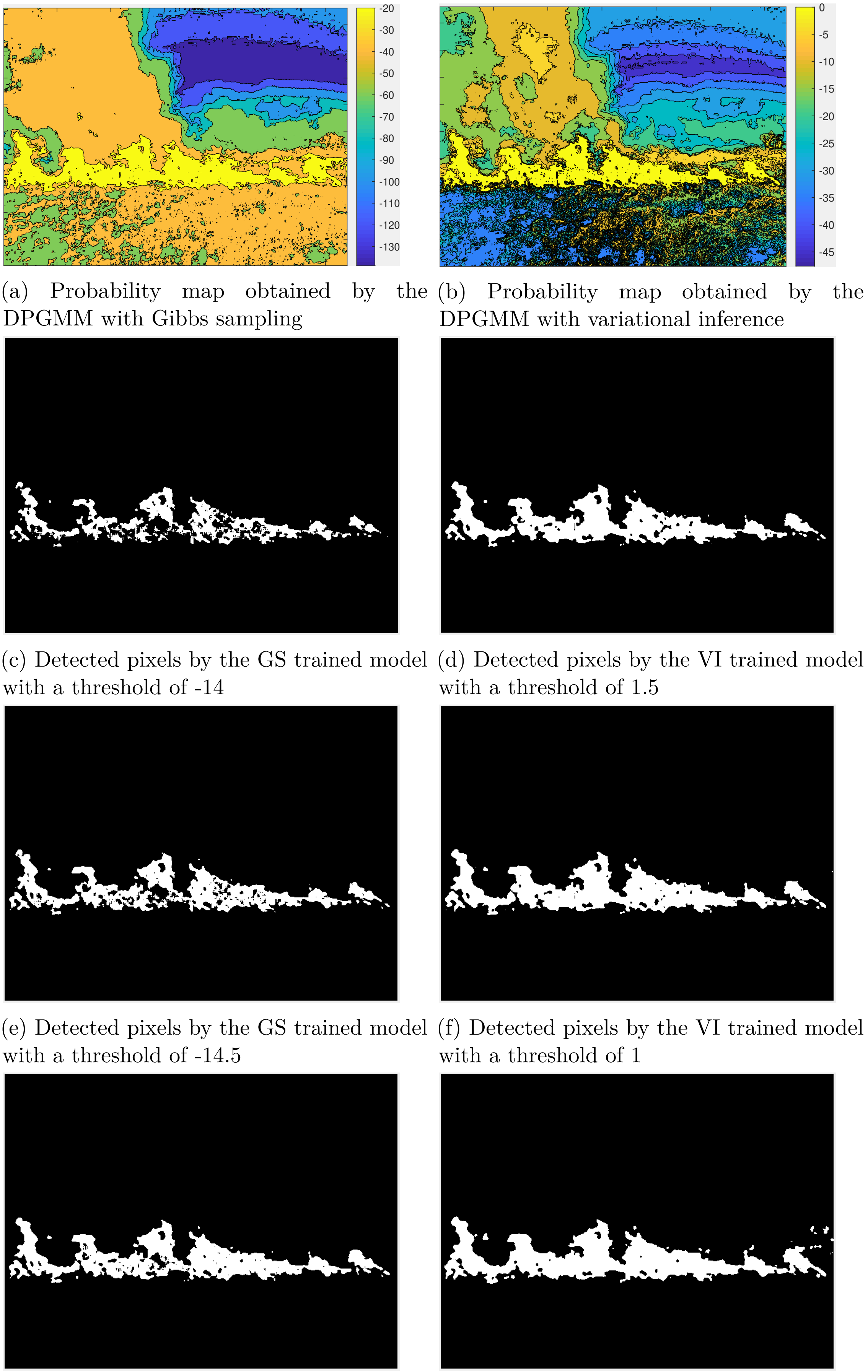


(a) Overall TPRs and TNRs of the framework using the proposed DPGMM based flame colour model with different thresholds of the estimated colour probability.

(b) Average processing time per frame of the framework using the proposed DPGMM based flame colour model with different thresholds of the colour probability.

Figure 4.13. Detection performance and computational complexity of the framework using the proposed DPGMM based flame colour model with different thresholds.

while many non-flame regions of quasi-flame colours can be effectively distinguished from real flames, such as the fire trunk in Figure 4.5 and the high-visibility clothing in Figure 4.6. Results obtained by the flame colour model trained with the GS algorithm are shown on the left columns of Figures 4.14 to 4.17 with the thresholds of flame probabilities set to -14, -14.5 and -15, respectively, while detected pixels by the trained model using the VI algorithm are provided on the right columns given the thresholds of 1.5, 1 and 0, respectively. More flame pixels will be successfully detected with a low threshold which also results in more falsely classified non-flame pixels, while a high threshold usually leads to fewer detected flame pixels and falsely detected non-flame ones. It is consistent with the ROC curves in Figure 4.2.



(a) Probability map obtained by the DPGMM with Gibbs sampling

(b) Probability map obtained by the DPGMM with variational inference

(c) Detected pixels by the GS trained model with a threshold of -14

(d) Detected pixels by the VI trained model with a threshold of 1.5

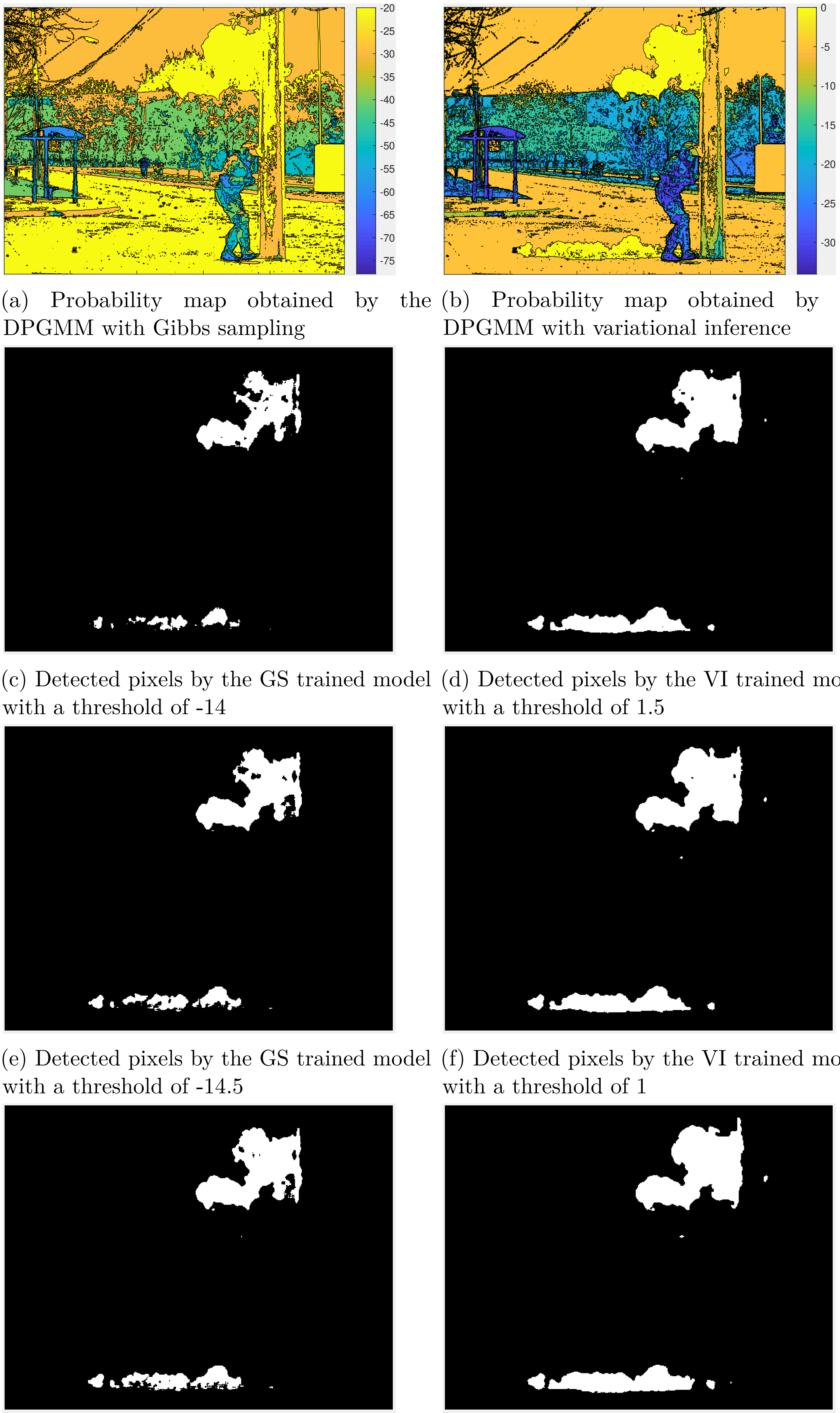
(e) Detected pixels by the GS trained model with a threshold of -14.5

(f) Detected pixels by the VI trained model with a threshold of 1

(g) Detected pixels by the GS trained model with a threshold of -15

(h) Detected pixels by the VI trained model with a threshold of 0

Figure 4.14. Probability maps obtained by the DPGMM based flame colour model with the GS and VI algorithms and the results with different thresholds of a testing image of flames in a forest fire in Figure 4.3.



(a) Probability map obtained by the DPGMM with Gibbs sampling

(b) Probability map obtained by the DPGMM with variational inference

(c) Detected pixels by the GS trained model with a threshold of -14

(d) Detected pixels by the VI trained model with a threshold of 1.5

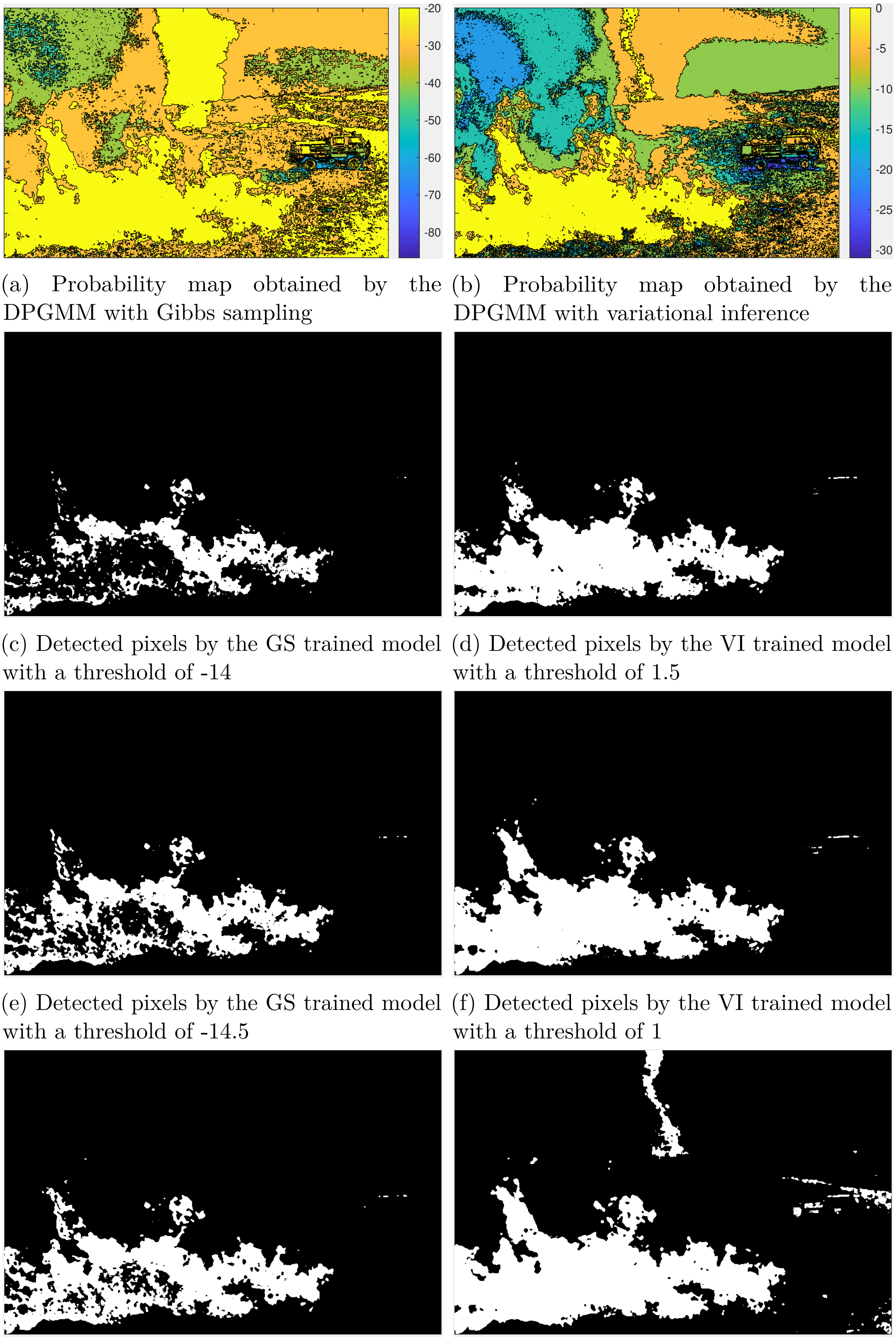
(e) Detected pixels by the GS trained model with a threshold of -14.5

(f) Detected pixels by the VI trained model with a threshold of 1

(g) Detected pixels by the GS trained model with a threshold of -15

(h) Detected pixels by the VI trained model with a threshold of 0

Figure 4.15. Probability maps obtained by the DPGMM based flame colour model with the GS and VI algorithms and the results with different thresholds, of a testing image of flames in a street in Figure 4.4.



(a) Probability map obtained by the DPGMM with Gibbs sampling

(b) Probability map obtained by the DPGMM with variational inference

(c) Detected pixels by the GS trained model with a threshold of -14

(d) Detected pixels by the VI trained model with a threshold of 1.5

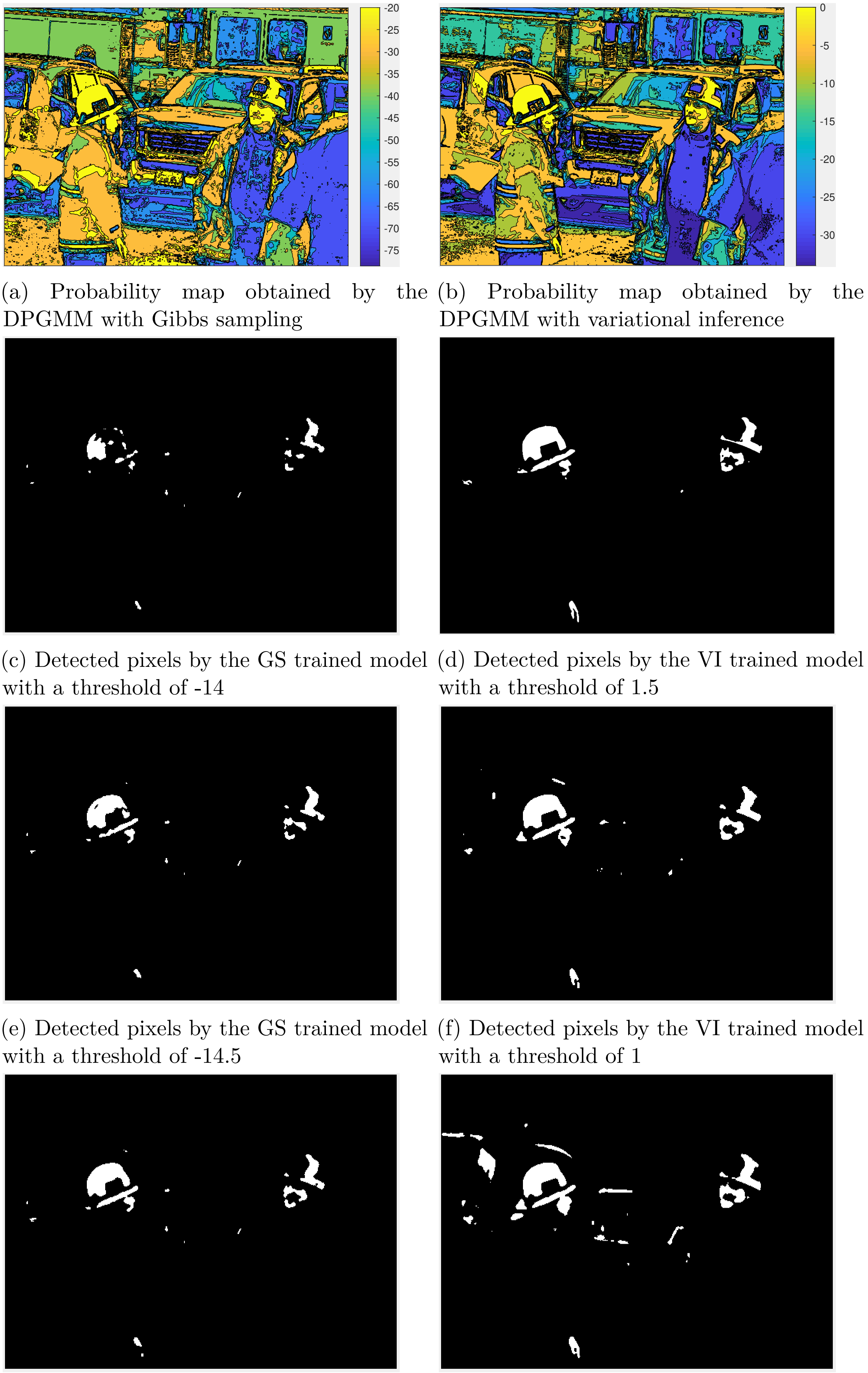
(e) Detected pixels by the GS trained model with a threshold of -14.5

(f) Detected pixels by the VI trained model with a threshold of 1

(g) Detected pixels by the GS trained model with a threshold of -15

(h) Detected pixels by the VI trained model with a threshold of 0

Figure 4.16. Probability maps obtained by the DPGMM based flame colour model with the GS and VI algorithms and the results with different thresholds, of a testing image of a fire truck and flames on branches in Figure 4.5.



(a) Probability map obtained by the DPGMM with Gibbs sampling

(b) Probability map obtained by the DPGMM with variational inference

(c) Detected pixels by the GS trained model with a threshold of -14

(d) Detected pixels by the VI trained model with a threshold of 1.5

(e) Detected pixels by the GS trained model with a threshold of -14.5

(f) Detected pixels by the VI trained model with a threshold of 1

(g) Detected pixels by the GS trained model with a threshold of -15

(h) Detected pixels by the VI trained model with a threshold of 0

Figure 4.17. Probability maps obtained by the DPGMM based flame colour model with the GS and VI algorithms and the results with different thresholds, of a testing image of fire trucks and fire fighters in high-visibility workwear without flames in Figure 4.6.

4.4 Summary

In this chapter, a novel model based on the DPGMM is proposed to approach the distribution of the colours of flames, which can be incorporated into the frameworks for automatic flame detection. The distribution of flame colours is modelled by a GMM whose prior is set to a DP. Instead of setting the unknown number of mixture components empirically, the proposed model learns it from training data, which contributes to an accurate estimation of the distribution of flame colours.

The inference of the proposed model is implemented by both the GS and VI algorithms. The GS algorithm is theoretically more accurate than the VI approach because it is based on the samples of the exact posterior. However, the computational complexity of the GS approach is high, and can not manage a large number of data. In contrast, the VI approach approximates the posterior with a family of variational distributions and turns the problem into an optimization one. The inference with the VI algorithm works more efficiently than the GS method. The improvement using a kd-tree further accelerates the inference, which enables the flame colour model to be trained with a large number of data and achieve good performance.

Experiments show that the proposed colour model outperforms other state-of-the-art models. Together with the saliency analysis on motion and the wavelet transform based temporal feature, the developed colour model contributes to accurate frame-wise detection of flames.

Chapter 5

Flame Detection Frameworks Based on Flame R-CNN and Faster R-CNN

It is challenging to achieve accurate detection of flames that are diverse in appearance and keep low false alarm rates in various environments at the same time. The objects that are similar to flames in colours or temporally changing patterns may cause frequent false alarms, which limits the practical applications of the frameworks of automatic flame detection.

To improve the performance of flame detection systems, two frameworks are proposed based on the Flame R-CNN and faster R-CNN in this chapter. In the former framework, a novel flame proposal generation scheme is designed by considering both the dynamic and colour properties of flames. Within it, the motion in videos can be accurately detected by the OR-PCA algorithm, even in a noisy environment. The flame proposals are generated by combining the OR-PCA algorithm and the flame colour model which has been introduced in Chapter 4. Subsequently, the generated flame proposals are projected onto a convolutional feature map to produce several small feature maps of fixed size using an RoI pooling layer, which will be further processed by additional layers. Regions of flames are outputted by the framework, based on which frame-wise results can be obtained. In the framework for flame detection based on faster R-CNN, the proposals are generated by a region proposal network which utilises the features produced by convolutional layers. The generated proposals are processed in a similar way to the framework of flame R-CNN. The framework of faster R-CNN outputs regions of flames as well.

5.1 Online Robust PCA via Stochastic Optimization

Foreground detection in videos separates moving objects from the stationary background in each frame. It plays a crucial role in flame detection in videos because flames usually change rapidly and wildly. Robust and effective foreground detection methods will discard the flame-coloured regions in the background, and thus reduce false alarms as well as the computational complexity of flame detection frameworks.

The RPCA algorithms decompose a data matrix into two parts, a low-rank matrix and a sparse matrix. Inspired by the setting, frames in videos that are resized into columns can be stacked into a matrix and processed by the RPCA schemes. The low-rank component naturally corresponds to the stationary background in a video, and the sparse matrix contains the information of moving objects. Compared with the classic PCA approaches [86], the RPCA algorithms are more robust to outliers and noise. Therefore, an algorithm of OR-PCA is embedded into the proposed framework of flame R-CNN and the details of the algorithm is introduced below.

Denote the matrix of observed data as $\mathbf{M} = \{\mathbf{m}_i\}_{i=1}^n, \mathbf{m} \in \mathbb{R}^d$, that can be decomposed as

$$\mathbf{M} = \mathbf{Y} + \mathbf{S}, \tag{5.1}$$

where $\mathbf{Y} = \{\mathbf{y}_i\}_{i=1}^n, \mathbf{y}_i \in \mathbb{R}^d$ and $\mathbf{S} = \{\mathbf{s}_i\}_{i=1}^n, \mathbf{s}_i \in \mathbb{R}^d$ represent the low-rank matrix and the sparse component, respectively, d denotes the dimension of each column, and n is the number of samples.

To accurately recover the low-rank matrix from corrupted observations, PCP [87], as one of the most widely used RPCA algorithms, solves the problem by

$$\min_{Y,S} \frac{1}{2}\|\mathbf{M} - \mathbf{Y} - \mathbf{S}\|_F^2 + \widetilde{\lambda}_1\|\mathbf{Y}\|_* + \widetilde{\lambda}_2\|\mathbf{S}\|_1, \tag{5.2}$$

where $\|\cdot\|_F^2$, $\|\cdot\|_*$, and $\|\cdot\|_1$ denote the Frobenius norm, nuclear norm and ℓ_1-norm of a matrix, respectively, and $\widetilde{\lambda}_1, \widetilde{\lambda}_2$ are balanced parameters. The nuclear norm of a matrix $A \in \mathbb{R}^{d\times n}$ is defined as the sum of its singular values $\{\sigma_i(A)\}_{i=1}^{\min\{d,n\}}$, i.e. $\|A\|_* = \sum_i \sigma_i(A)$ [88].

However, the PCP methods work in a batch manner and need to access all samples in each iteration of the optimization, which requires large storage for data and results in delay in processing [89]. The main problem which prevents the PCP algorithm to work online is that the nuclear norm couples all of the samples. To solve this problem, Feng and Xu proposed an algorithm by employing an equivalent form of the nuclear norm [90], which is given by

$$\|\mathbf{Y}\|_* = \inf_{\mathbf{L}\in\mathbb{R}^{d\times r},\mathbf{R}\in\mathbb{R}^{n\times r}} \left\{\frac{1}{2}\|\mathbf{L}\|_F^2 + \frac{1}{2}\|\mathbf{R}\|_F^2 : \mathbf{Y} = \mathbf{L}\mathbf{R}^\mathsf{T}\right\}, \tag{5.3}$$

where r denotes upper bound of the rank of $\mathbf{Y}$, and $\inf\{\cdot\}$ represents the infimum of a set. The low rank matrix is factorized into two parts $\mathbf{L} \in \mathbb{R}^{d\times r}$ and $\mathbf{R} \in \mathbb{R}^{n\times r}$, where $\mathbf{L}$ can be treated as basis of the low dimensional subspace and $\mathbf{R}$ are coefficients of samples projected to the basis. Therefore, the problem in (5.2) can be rewritten as

$$\min_{\mathbf{L}\in\mathbb{R}^{d\times r},\mathbf{R}\in\mathbb{R}^{n\times r},\mathbf{S}} \frac{1}{2}\|\mathbf{M}-\mathbf{L}\mathbf{R}^{\mathsf{T}}-\mathbf{S}\|_F^2+\frac{\widetilde{\lambda}_1}{2}(\|\mathbf{L}\|_F^2+\|\mathbf{R}\|_F^2)+\widetilde{\lambda}_2\|\mathbf{S}\|_1. \tag{5.4}$$

It can be proven that the local minima of (5.4) are global minima of (5.2) [90]. Therefore, solving the problem in (5.4) will provide estimations of the low dimensional subspace and sparse component equivalent to those obtained by (5.2).

Optimizing the objective function in (5.4) is equivalent to minimizing the empirical cost function with finite samples given by

$$f_n(\mathbf{L}) \triangleq \frac{1}{n}\sum_{i=1}^{n}\ell(\mathbf{m}_i,\mathbf{L})+\frac{\widetilde{\lambda}_1}{2n}\|\mathbf{L}\|_F^2, \tag{5.5}$$

where the loss function $\ell(\mathbf{m}_i,\mathbf{L})$ for the sample $\mathbf{m}_i$ is defined by

$$\ell(\mathbf{m}_i,\mathbf{L}) \triangleq \min_{\mathbf{r}_i,\mathbf{s}_i}\frac{1}{2}\|\mathbf{m}_i-\mathbf{L}\mathbf{r}_i-\mathbf{s}_i\|_2^2+\frac{\widetilde{\lambda}_1}{2}\|\mathbf{r}_i\|_2^2+\widetilde{\lambda}_2\|\mathbf{s}_i\|_2^2. \tag{5.6}$$

The loss function measures the representation error of a sample projected onto fixed basis $\mathbf{L}$. When using stochastic optimization, it is more important to minimize the expected cost over all observations defined by

$$f(\mathbf{L}) \triangleq \mathbb{E}_{\mathbf{m}}[\ell(\mathbf{m},\mathbf{L})] = \lim_{n\to\infty} f_n(\mathbf{L}), \tag{5.7}$$

where the expectation is taken with respect to $\mathbf{m}$.

The method aims at estimating the low dimensional subspace and the sparse component in an online manner. It is designed to process each new sample once it is obtained without accessing the observations after it. It updates the coefficients $\mathbf{r}$ as well as the sparse component $\mathbf{s}$ of the new sample, and the subspace basis $\mathbf{L}$ alternatively using a stochastic optimization algorithm. Specifically, the coefficients $\mathbf{r}_t$ and sparse component $\mathbf{s}_t$ at time t are obtained by minimising the loss function in Eq. (5.6) with the basis $\mathbf{L}_{t-1}$ of time $t-1$ fixed. Subsequently, the basis $\mathbf{L}_t$ are estimated by by minimizing the cumulative loss with respect to $\{\mathbf{r}_i\}_{i=1}^t$ and $\{\mathbf{s}_i\}_{i=1}^t$. The process is conducted for several iterations. To update the basis, the OR-PCA algorithm creates and optimizes a surrogate function defined as follows for the expected cost $f(\mathbf{L})$

$$g_t(\mathbf{L}) \triangleq \frac{1}{t}\sum_{i=1}^{t}\left(\frac{1}{2}\|\mathbf{y}_i - \mathbf{L}\mathbf{r}_i - \mathbf{s}_i\|_2^2 + \frac{\widetilde{\lambda}_1}{2}\|\mathbf{r}_i\|_2^2 + \widetilde{\lambda}_2\|\mathbf{s}_i\|_1\right) + \frac{\widetilde{\lambda}_1}{2t}\|\mathbf{L}\|_F^2 \, . \tag{5.8}$$

It can be proven that the proposed OR-PCA converges to the optimal solution provided by the PCP algorithm which is in a batch manner.

5.2 Flame Detection Framework Based on Faster R-CNN

The methods of CNNs have obtained significant success in many computer vision related tasks. Several CNNs of different architectures have achieved good performance in both object and scene classification if given enough training data [65, 67]. However, they do not work effectively when applied to the task of flame detection directly because of the diverse appearance of flames and complicated environments of fires.

A large number of images with and without flames are needed to train a CNN for accurate results of classification because of the great diversity of flames in appearance. However, the available training images are limited and imbalanced, as there are much more images without flames than those containing flames. Additionally, flames sometimes occupy only a small part of the scene in videos if fires happen in distant places. In such a situation, the features of flames may be overwhelmed by those of a cluttered background, which will confuse the CNNs and thus influence the performance. Therefore, it is reasonable to assume that processing candidate regions of flames can improve the results instead of conducting classification on entire images using CNNs.

Methods of object detection include two main branches: two-stage and single-stage approaches [91]. The former group includes the widely known R-CNN [92] and its improved versions [93–95], while the later group has the famous YOLO detector [96] and its upgraded methods [97, 98]. Single-stage methods usually have lower computational complexity than two-stage frameworks at the expense of a decrease in accuracy. Because of the huge losses caused by fires, the accuracy of flame detection methods plays a more important role than efficiency. Therefore, a framework for flame detection is proposed based on the faster R-CNN, which is among the two-stage methods [94].

The diagram of the framework is shown in Figure 5.1. The network takes a frame of videos and a set of flame proposals produced by a region proposal network as input. The input frame is processed by several convolutional layers (as well as max pooling and ReLU layers) to produce a feature map. The generated flame

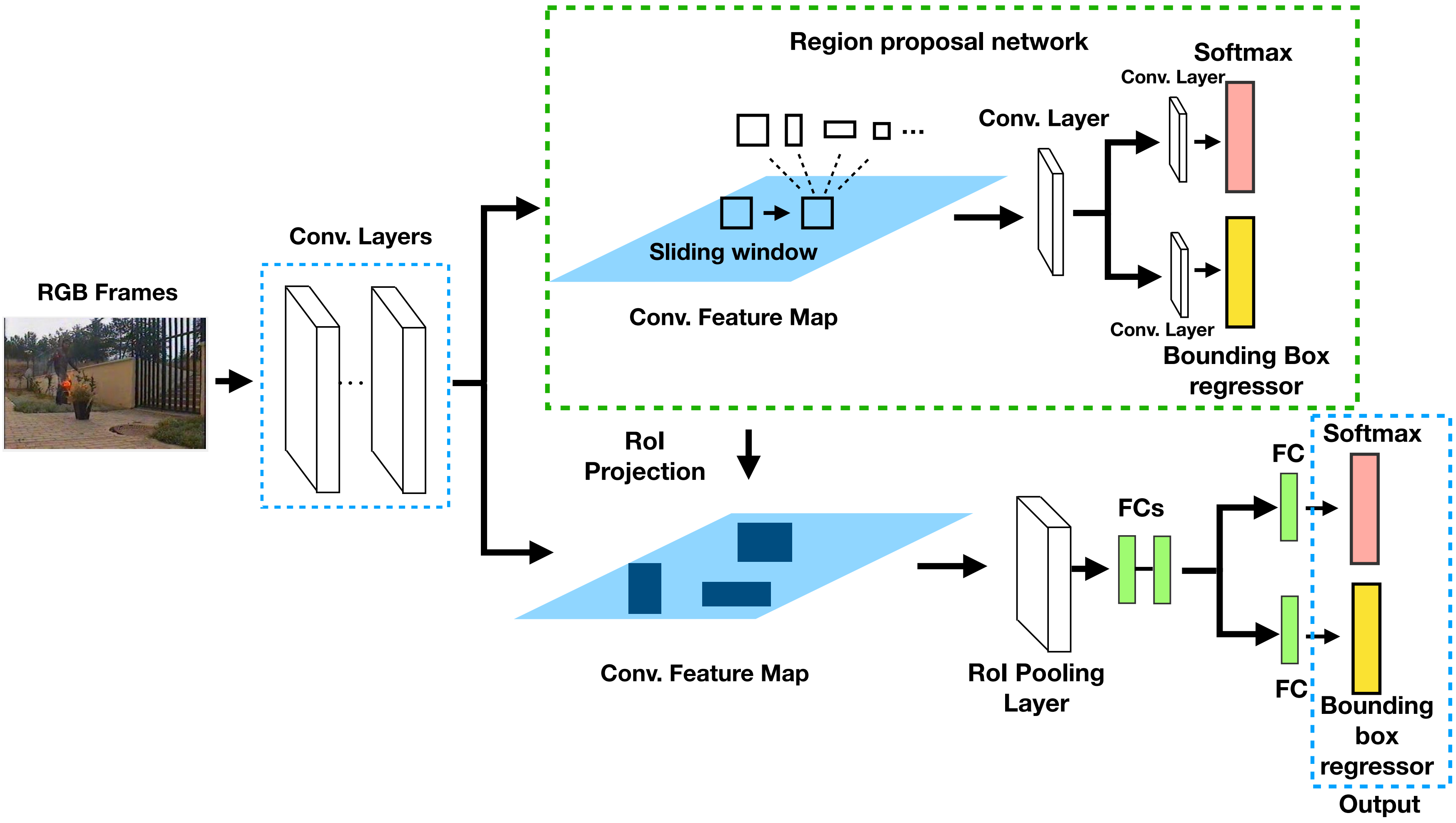


Figure 5.1. Diagram of the framework of faster R-CNN for flame detection.

proposals are projected onto the convolutional feature map and produce several small feature maps of a fixed size based on an RoI pooling layer. The obtained small maps will be processed by several fully-connected (fc) layers to generate feature vectors which are subsequently fed into two sibling output layers, i.e., a classification layer and a bounding box regression layer. The classification layer provides estimated probabilities of flames and the background, while the bounding box regression layer outputs the locations of the bounding boxes of flames.

5.2.1 RoI Pooling Layer

The RoI pooling layer transfers the convolutional feature map together with the generated proposals of different spatial sizes into small feature maps of the same size by max pooling, so that these small maps can be processed by fully connected layers. Given a proposal of height h_p and width w_p, the RoI layer divides it into a $H_p \times W_p$ grid of sub-windows and performs max-pooling in each of the sub-window, of which the approximate size is $h_p/H_p \times w_h/W_h$ (the H_p and W_p are preset hyperparameters). In this way, the RoI pooling layer outputs a small feature map of size $H_p \times W_p$ for each proposal generated by the region proposal network.

5.2.2 Multi-task Loss of the Detection Network

The detection network outputs a probability distribution p_c and a bounding box regression offset $b_f = (b_x, b_y, b_w, b_h)$ for each proposal of flames, where $c \in \{0, 1\}$ is the variable indicating the classes of proposals. Specifically, $c = 0$ and $c = 1$ correspond to the background and flames, respectively, while b_f denotes the offset of the outputted bounding box relative to the proposal generated by the region proposal network. For each proposal, a multi-task loss is proposed to consider both classification and bounding box regression as follows

$$\mathcal{L}(p_c, c^*, b_f, b_f^*) = L_{cls}(p_c, c^*) + \lambda' \mathbb{I}(c^* = 1) L_{loc}(b_f, b_f^*), \tag{5.9}$$

where $c^* \in \{0, 1\}$ denotes the ground truth class of the proposal, b_f^* represents the ground truth regression offset of the bounding box of flames, and λ' is a hyperparameter to balance the losses of classification and regression. The term $\mathbb{I}(c^* = 1)$ indicates that the loss of the bounding box regression is taken into account only if the proposal is classified as flames, where $\mathbb{I}(\cdot)$ is the indicator function defined in Eq. (4.5). The loss of the classification task is defined as $L_{cls}(p_c, c^*) = -\log p_c(c^*)$, and the loss of bounding box regression is measured by a smooth L_1 loss as

$$L_{loc}(b_f, b_f^*) = \sum_{j \in \{x,y,w,h\}} \text{smooth}_{L_1}(b_j, b_j^*), \tag{5.10}$$

where the smooth L_1 loss is defined as

$$\mathrm{smooth}_{L_1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise} \end{cases} . \tag{5.11}$$

The positive ($c^* = 1$) and negative ($c^* = 0$) training proposals are those that have IoU overlap with any ground truth bounding box of flames in the intervals $[0.5, 1]$ and $[0.05, 0.2)$, respectively.

5.2.3 Region Proposal Network

The flame proposals are generated by a region proposal network which shares a set of convolutional layers with the detection network to enhance the computational efficiency. A small sliding network is placed over the convolutional feature map outputted from the last shared convolutional layer to generate probable regions of objects. Specifically, the sliding network is fully connected to a rectangular window of size $n \times n$ of the convolutional feature map. The window is mapped to a vector which is fed into two parallel fully-connected layers, i.e. a softmax classification layer and a regression layer. The classification layer outputs scores of each proposal containing objects and belonging to the background, while the regression layer provides refined bounding boxes. As the small network slides across the entire image, the weights of the fully-connected layers are shared across all the locations. Therefore, the network can be implemented by an $n \times n$ convolutional layer followed by two 1×1 convolutional layers, as shown in Figure 5.1. Additionally, for each location of the sliding window, the region proposal network outputs k bounding boxes. The coordinates and sizes of the predicted proposals are relative to k anchor boxes of multiple scales and aspect ratios, which are centred at the sliding window. Each anchor box is assigned to a binary label to indicate if it contains an object. An anchor box will be labelled as positive, i.e. the ground truth class variable $\tilde{c}^* = 1$, if its intersection over union (IoU) overlap with a ground truth bounding box is the highest or higher than 70%. Otherwise, it will be assigned a negative label. It is noteworthy that $\tilde{c}^*$ is different from the ground truth label c^* of proposals introduced in Section 5.2.2, which indicates if a generated proposal contains flames in it.

A multi-task loss is minimised to train the region proposal network, which is defined by

$$\mathcal{L}(\{p^i_{\tilde{c}}\}, \{b^i_a\}) = \frac{1}{N_{cls}} \sum_i L_{cls}\big(p^i_{\tilde{c}}, (\tilde{c}^*)^i\big) + \frac{\tilde{\lambda}}{N_{reg}} \sum_i \mathbb{I}\big((\tilde{c}^*)^i = 1\big) L_{reg}\big(b^i_a, (b^*_a)^i\big), \tag{5.12}$$

where i is the index of an anchor box, $p_{\tilde{c}}^i$ and $(\tilde{c}^*)^i$ denote the probability distribution and the ground truth label of the i-th anchor box, respectively, and $\tilde{\lambda}$ is a hyperparameter which balances the losses of classification and regression. The predicted and the ground truth bounding boxes associated with the i-th anchor box are denoted by b_a^i and $(b_a^*)^i$, respectively. The loss of classification is measured as $L_{cls}\big(p_{\tilde{c}}^i, (\tilde{c}^*)^i\big) = -\log p_{\tilde{c}}^i\big((\tilde{c}^*)^i\big)$, while the loss of regression is measured by the smooth L_1 loss function defined in Eq. (5.10) and Eq. (5.11). The losses are normalised by N_{cls} and N_{reg}, which are set to the mini-batch size and the number of anchor box locations, respectively.

5.2.4 Training of Faster R-CNN

The training of the faster R-CNN is implemented by an approach of four steps:
1. The region proposal network is fine-tuned on a pre-trained network for the task of proposal generation;
2. A separate detection network is trained using the framework of fast R-CNN [93] with the proposals generated by the region proposal network obtained in step 1;
3. The detection network trained in step 2 is used to initialise and fine-tune the region proposal network, with the shared convolutional layers fixed, which means only the layers unique to the region proposal network are changed in this step;
4. The layers unique to the detection network are fine-tuned when fixing the shared convolutional layers.

In this way, the training of the entire faster R-CNN is implemented.

5.3 Flame Detection Framework Based on Flame R-CNN

Most targets of the tasks of object detection are rigid objects, such as vehicles, pedestrians and animals, which have limited diversity in appearance within each class. Different from rigid objects, flames have a rich variety of shapes and colours, as shown in the sample images in Figure 2.2, which makes it challenging to generate proposals of flames in the same way as the frameworks of rigid object detection, e.g. the selective search algorithm [92, 93, 99]. In the two-stage frameworks of object detection, RoIs of objects are selected in the first stage, based on which the classification and bounding box regression are conducted subsequently. Therefore, the generation of proposals has a great influence on the performance of the flame detection framework. Specifically, failing to generate proposals over regions of flames will lead to accuracy degeneration. Therefore, it is necessary to improve the method of flame proposal generation.

A novel framework of flame R-CNN is proposed in this section for the task of flame detection, in which the proposals are generated based on features describing the characteristics unique to flames. As mentioned in Chapter 2, the colours and dynamics are two crucial properties of flames. Therefore, a flame proposal generation scheme is proposed based on these two properties.

The diagram of the framework is illustrated in Figure 5.2. It can be seen that the framework of flame R-CNN takes each frame of videos as input, and outputs regions containing flames. A frame-wise decision can be made according to those detected regions of flames. The input frame is processed by several convolutional layers (as well as ReLU layers and max pooling layers) to produce a convolutional feature map. Simultaneously, flame proposals are generated by the flame proposal generation scheme based on the colour and dynamic properties of flames. The generated proposals are subsequently projected to the corresponding locations on the convolutional feature map. Based on an RoI pooling layer, the features inside each generated flame proposal over the feature map are transferred into a small map of a fixed size, which can be further processed by fully connected layers and a softmax classification layer. The framework outputs the proposals which are classified as flames.

5.3.1 Flame Proposal Generation Scheme

Flames differ from rigid objects in many ways, which influences their detection. One of the crucial differences is that some flames, especially weak ones, are of semi-transparent colours, which induces unclear edges of flames and visibility of the objects behind fires. It makes it more difficult to generate proposals of flames than rigid objects. In order to solve this problem, the proposed flame proposal generation scheme generates proposals based on the dynamic and colour properties of flames, which are effective in distinguishing flames from the background.

On one hand, the changes in the intensity values of probable flame regions can be detected by the OR-PCA algorithm introduced in Sec. 5.1 in an online manner. On the other hand, potential pixels of flames can be selected by the DPGMM based colour model proposed in Chap.4. The combination of the OR-PCA algorithm and DPGMM colour model is capable of detecting candidate pixels of flames, which will contribute to the generation of flame proposals. Flame proposals are generated based on a grid of boxes, as shown in Figure 5.2. In the diagram, the boxes are non-overlapping for visualisation, but they overlap in actuality. The boxes, in which the ratios of candidate flame pixels are higher than a threshold τ_f, are treated as flame proposals. It is noteworthy that the boxes can be multi-scaled, but of a fixed aspect ratio to preserve the texture-related features of the regions within the proposals. Specifically, each flame proposal is projected onto the convolutional

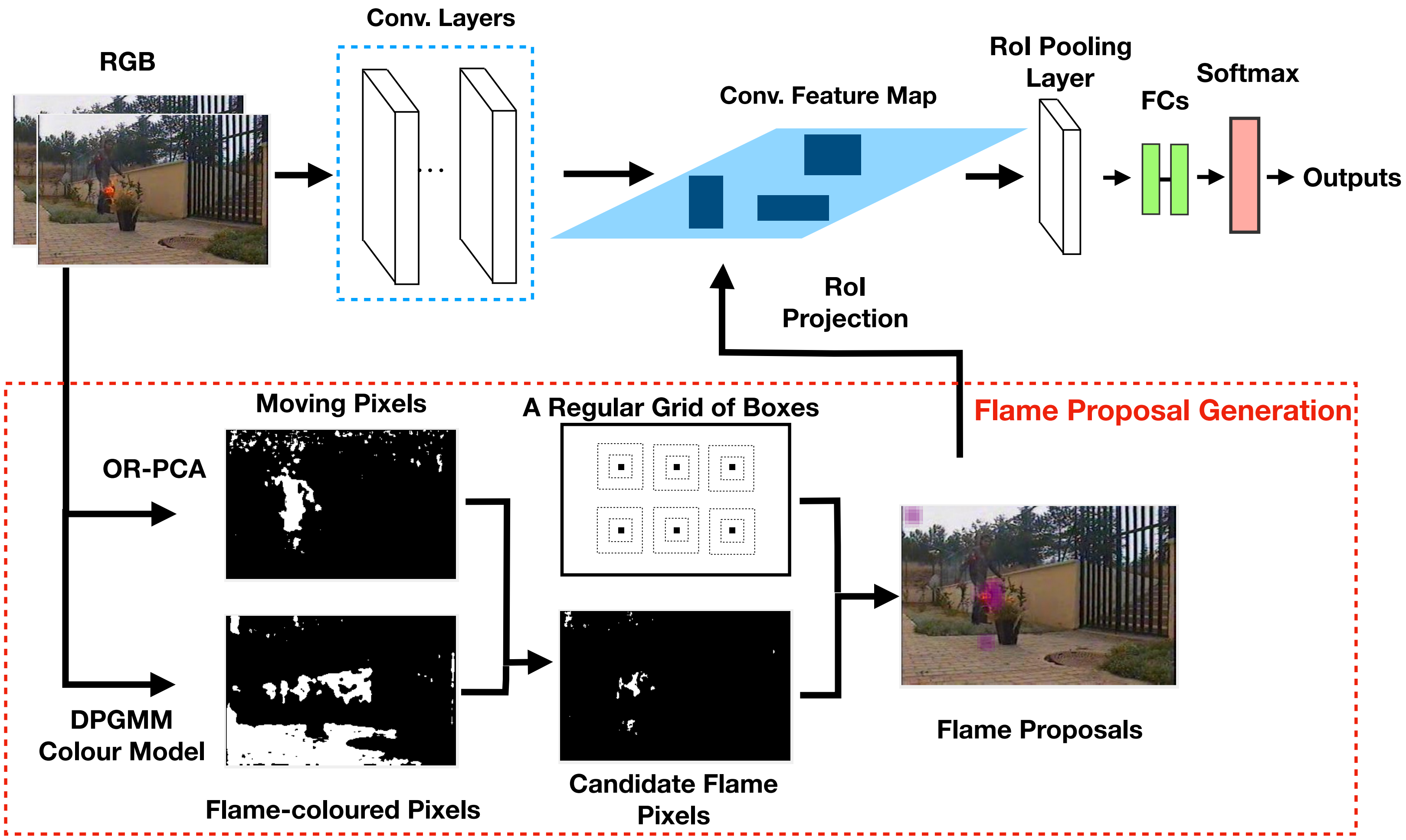


Figure 5.2. Diagram of the framework based on flame R-CNN.

feature map to generate a small feature map, which will be resized by the RoI layer. Consequently, the texture-related features will be changed if the aspect ratio of the proposal is significantly different from the fixed size of the output of the RoI layer. This will influence the performance of the framework. It also explains the reason why the flame proposals are not generated by the selective search approach [99] or connected components labelling operator [100]. The proposals generated in these ways are diverse in aspect ratios resulted from the various appearance of flames, which will significantly change the texture-related features when being processed by the RoI layer. As mentioned in Section 2.1.1, the texture-related features are of crucial importance in describing flames, so the influence on these features of resizing operation should be avoided. Additionally, the grid of boxes can be set at a fine level and generate multi-scale proposals, which can improve the detection of flames occupying small regions in the scene.

Besides videos, the proposed scheme can also work with images by selecting the candidate flame pixels only based on the colour model.

5.3.2 Loss Function of the Framework of Flame R-CNN

As mentioned above, the flame proposals are generated based on a regular grid of boxes. They are not designed to bound each flame region with a bounding box, which is a common setting in rigid object detection. Instead, the proposed framework aims to cover as many flame regions as possible. Therefore, it is unnecessary to perform the regression of boxes and accordingly only the classification loss is considered. It is different from existing methods for object detection, such as fast R-CNN [93] and YOLO [96]. For each proposal of flames, the loss function of the proposed framework is given by

$$\mathcal{L}(p_c, c^*) = -\log p_c(c^*), \tag{5.13}$$

where c^* is the ground truth class label of the proposal, and p_c is the probability distribution outputted by the softmax layer.

5.3.3 Training of Flame R-CNN

The ground truth RoIs of flames are given in the format of a grid of boxes instead of the tightest bounding boxes. The boxes are superimposed on the training images. For each box, it is labelled as ’flame’ if the ratio of ground truth flame pixels is higher than 30%. The positive training proposals (containing flames) are those which have IoU overlap with any ground truth box of flames of at least 50%, while the ones with IoU overlap with a ground truth box within the range $[0.05, 0.2)$ are treated as negative proposals (non-flame).

5.4 Experimental Results and Discussion

5.4.1 Benchmarking Database and Experimental Settings

The proposed frameworks for flame detection based on faster R-CNN and flame R-CNN are trained on 729 images from the datasets [20, 84], and tested on 16 videos of 3968 frames from [73, 74]. The details of the datasets have been introduced in Sec. 3.5.1 with a brief description of the testing videos in Table 3.1. The proposed frameworks are trained on images instead of videos because the frames from the same video are similar and may induce the problem of over-fitting. Training the networks with images which are different from the testing videos can avoid this problem, and prove the robustness of the frameworks. Both the frameworks of flame R-CNN and faster R-CNN are fine-tuned based on a Resnet50 [67], which is pre-trained on the database for the ImageNet Large-Scale Visual Recognition Challenge [101], which is a subset of the ImageNet database [68].

The DPGMM based colour model in the framework of flame R-CNN is trained on 100 images randomly selected from [84]. The colour model is trained by VI instead of the GS, because of the better performance indicated by the results in Section 4.3.2. The proposed frameworks are evaluated by the frame-wise TPR and TNR, which have been introduced in Eqs. (3.27) and (3.28), and are compared with a state-of-the-art approach based on SqueezeNet [2].

5.4.2 Performance of Faster R-CNN for Flame Detection

The faster R-CNN for flame detection is trained using the Adam algorithm [102], of which the initial learning rate is set to 0.0001.

5.4.2.1 Anchor Box Estimation

In the framework of faster R-CNN for flame detection, the sizes of anchor boxes are crucial to the performance. Anchors that closely represent the scales and aspect ratios of flames will help to enhance the accuracy of detection. Specifically, the sizes of anchor boxes should be close to the sizes of the ground truth bounding boxes of flames. To properly set anchors, the sizes of the ground truth bounding boxes from the 729 training images are plotted in Figure 5.3. The wide diversity can be seen in both the area and aspect ratio of the bounding boxes. To choose proper sizes of anchor boxes, the k-means clustering algorithm is utilized with an IoU based distance metric instead of the Euclidean distance metric. As such, the sizes of anchor boxes can be estimated according to the training data given the number of anchors.

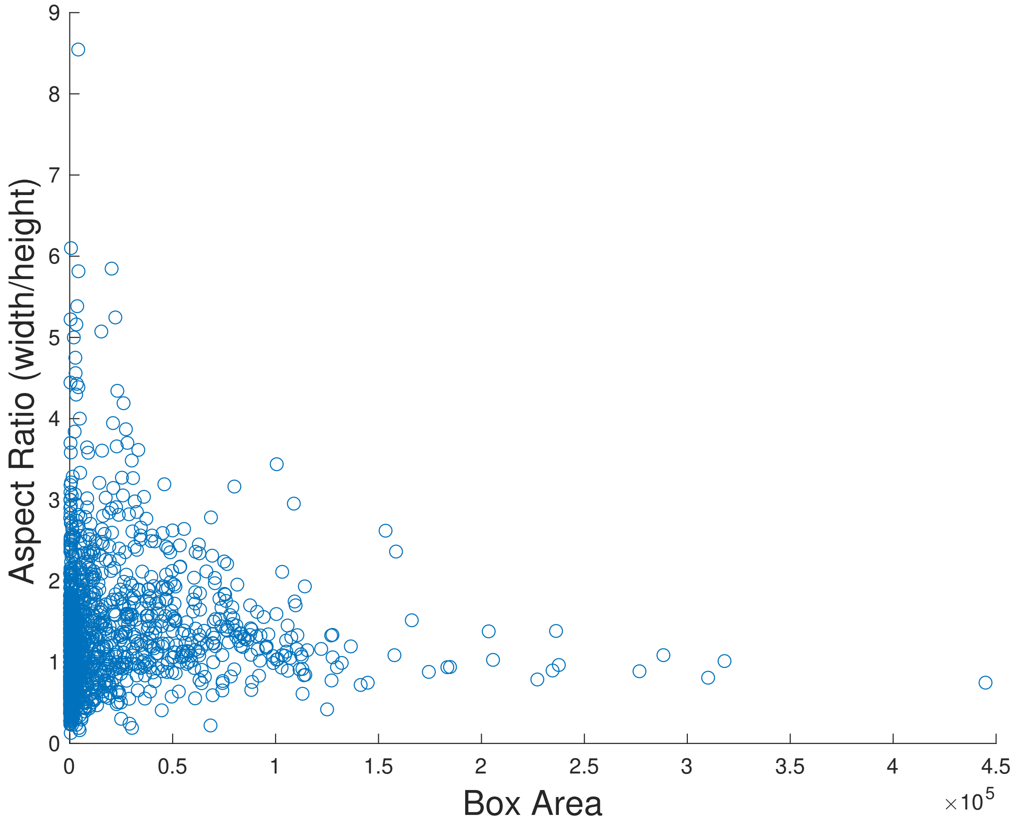


Figure 5.3. The sizes of the ground truth bounding boxes of flames in training images.

5.4.2.2 Results of Flame Detection in Videos

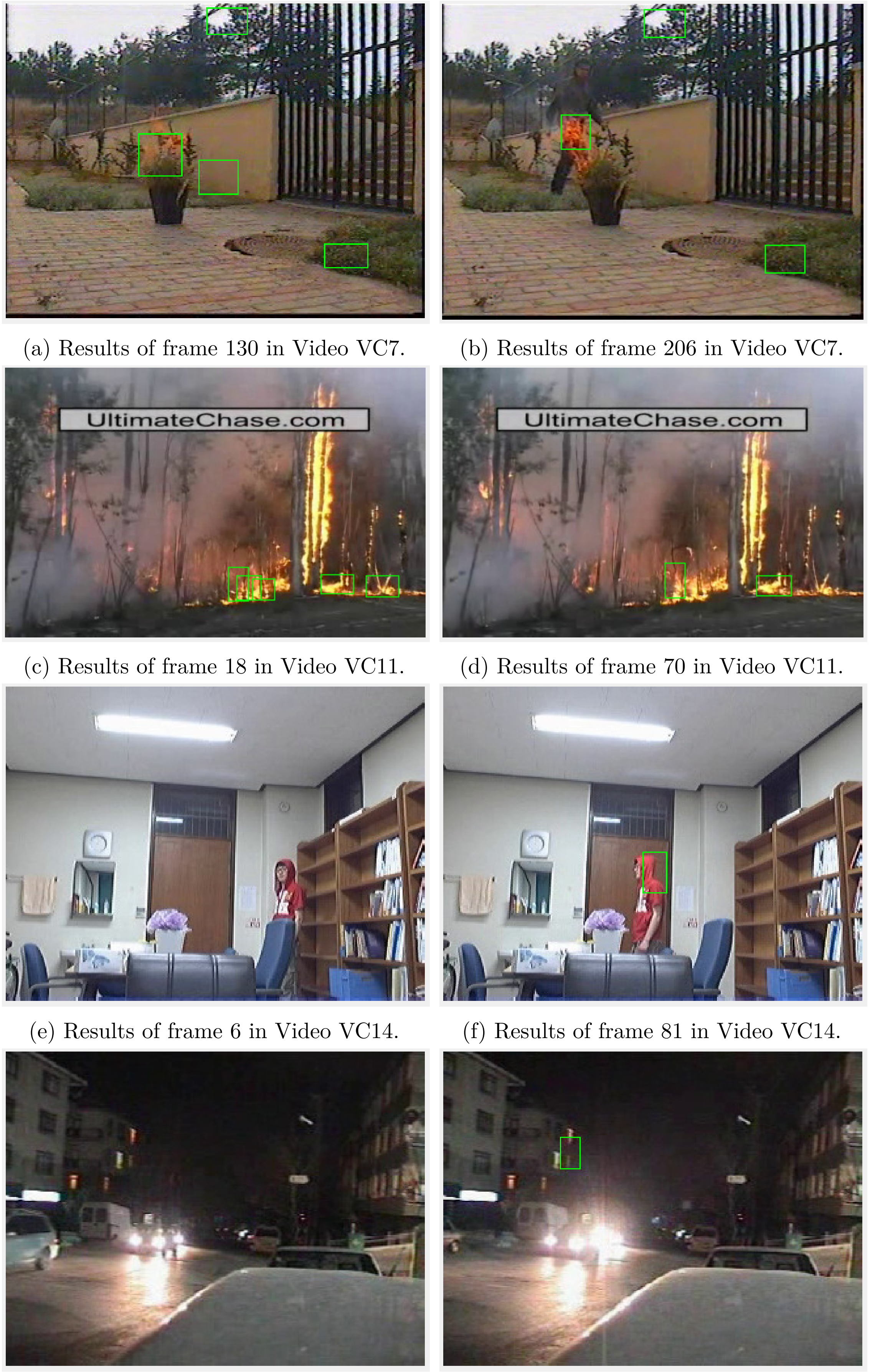
The number of anchors will influence the estimated sizes of anchor boxes, and thus may impact the performance of the framework of faster R-CNN for flame detection. To explore the influence, the framework of faster R-CNN is trained and tested using the anchor boxes that are estimated with the anchor number set to 3, 4, and 5, respectively. The sizes of the estimated anchor boxes are listed in Table 5.1 together with the performance of the framework. From the results illustrated in Table 5.1, it can be seen that the sizes of anchor box have considerable influence on the performance of the framework. Additionally, the average processing time of the framework using anchors of 3 different sizes is much less than that consumed by the faster R-CNN with 4 and 5 sizes of anchor boxes, which may be caused by the different numbers of generated proposals. Usually, the number of proposals increases with the number of anchors and thus leads to increased time to process each frame. However, the time of the frameworks using 4 and 5 sizes of anchors are almost the same. It is probably because one of the 5 sets of anchor boxes generates only a small number of proposals.

Table 5.1. Sizes of anchor boxes and corresponding performance of the framework of faster R-CNN for flame detection.

Number of sizes of anchors	3	4	5
Estimated sizes	30× 30 96× 125 229× 296	24× 24 63× 75 142× 197 275× 336	23× 23 54× 58 107× 128 164× 258 313× 350
Overall TPR	0.7040	0.8499	0.4851
Overall TNR	0.3007	0.7299	0.7823
Average processing time per frame/s	0.27	0.43	0.42

Table 5.2. Detection performance of the framework of faster R-CNN

Videos	*tp*	*fn*	*tn*	*fp*	Total positive frames	Total negative frames	TPR	TNR
VC1	26	0	0	0	26	0	1	-
VC2	93	0	0	0	93	0	1	-
VC3	48	0	0	0	48	0	1	-
VC4	34	7	0	0	41	0	0.8293	-
VC5	210	4	0	0	214	0	0.9813	-
VC6	43	133	0	0	176	0	0.2443	-
VC7	684	3	0	5	687	5	0.9956	0
VC8	563	9	0	69	572	69	0.9843	0
VC9	319	67	0	0	386	0	0.8264	-
VC10	194	201	0	0	395	0	0.4911	-
VC11	186	0	0	0	186	0	1	-
VC12	0	0	34	105	0	139	-	0.2446
VC13	0	0	63	81	0	144	-	0.4375
VC14	0	0	150	5	0	155	-	0.9677
VC15	0	0	334	44	0	378	-	0.8836
VC16	0	0	254	0	0	254	-	1
Overall	2400	424	835	309	2824	1144	0.8499	0.7299



(a) Results of frame 130 in Video VC7.

(b) Results of frame 206 in Video VC7.

(c) Results of frame 18 in Video VC11.

(d) Results of frame 70 in Video VC11.

(e) Results of frame 6 in Video VC14.

(f) Results of frame 81 in Video VC14.

(g) Results of frame 12 in Video VC12.

(h) Results of frame 20 in Video VC12.

Figure 5.4. Detected flame regions of the framework of faster R-CNN tested on sample frames of Video VC7, VC11, VC12, and VC14.

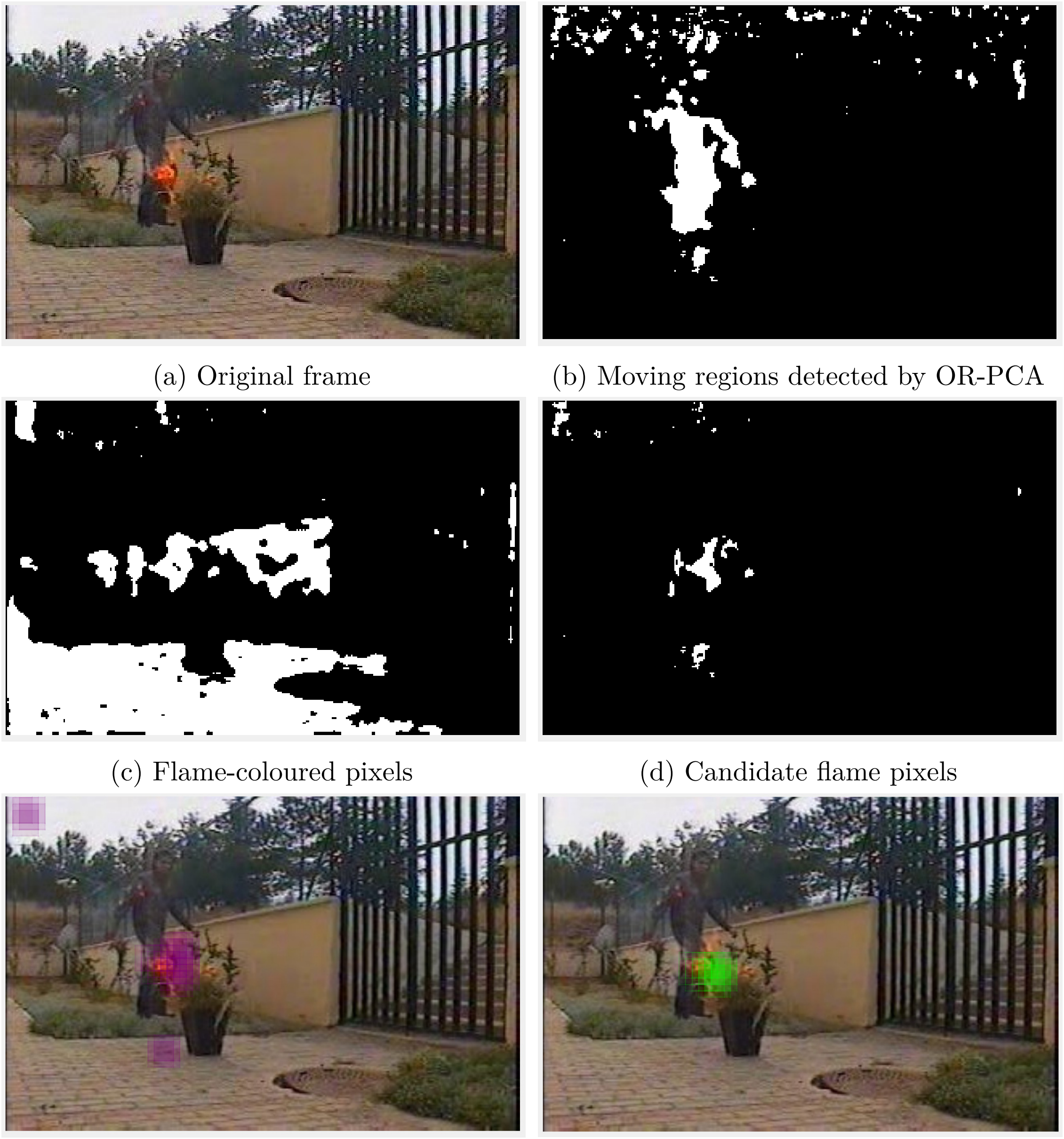
Table 5.2 shows the results of the framework using faster R-CNN with four sizes of anchors tested on each video. The results show that the TPRs and TNRs of some videos are much lower than those of others. To explore the reasons, detected flames on a few frames of testing videos are illustrated in Figure 5.4 by the framework of faster R-CNN with 4 anchors. Figure 5.4a and Figure 5.4b show that the framework successfully detects the flames of semi-transparent colours. However, many non-flame regions are falsely detected as flames apart from the real ones. Most of these regions are leaves or grass, which often appear in the training images together with flames. The network may falsely take the features of these objects as the features of flames. Furthermore, the framework fails to detect flames, of which the sizes do match any of the anchor boxes. For example, it does not detect the flames burning on the trunk of the tree in the middle of the frames in Figure 5.4c and Figure 5.4d. For the negative videos without flames, the framework of faster R-CNN achieves good performance in the videos which contains rigid objects of flame colours, while does not work so well in the videos of flashing car lights. It means the framework of faster R-CNN is suitable for classifying objects of relatively fixed shapes and can achieve accurate detection when the flames match one of the anchor boxes.

5.4.3 Detection Performance of Flame R-CNN

In the experiment of the flame R-CNN in this section, the sizes of boxes are set to be 16×16 with a stride of 4. The threshold τ_f for the ratio of candidate flame pixels is set to 0.3 in the flame proposal generation scheme. A threshold of the logarithmic probability of flame colours is set to -1.2 in the proposed framework. The threshold is set to a relatively low value to enhance the performance of proposal generation, which will influence the final detection results. In the flame proposal generation scheme, a low threshold of flame colour probabilities usually leads to more proposals than a high threshold. Once there is a flame without any generated proposal, a failure in detection will happen, which may cause huge losses. In contrast, a proposal with no flame in it will be further refined by additional layers within the framework of flame R-CNN and can produce a reliable result of the detection. Therefore, the threshold is set to -1.2 to improve the detection performance of the flame R-CNN. Additionally, the flame R-CNN is fine-tuned with ground truth flame proposals using the Adam algorithm [102], of which the initial learning rate is set to 0.0001.

The intermediate results, as well as the final detection performance of some of the testing videos, are illustrated in this section. From them, it can be seen that the framework of flame R-CNN achieves accurate detection of flame regions.

The effectiveness of the proposed flame proposal generation scheme can be proven by the results in Figure 5.5 and Figure 5.6. The OR-PCA algorithm works effectively in detecting the moving foreground objects while ignores the noise in the background.



(a) Original frame

(b) Moving regions detected by OR-PCA

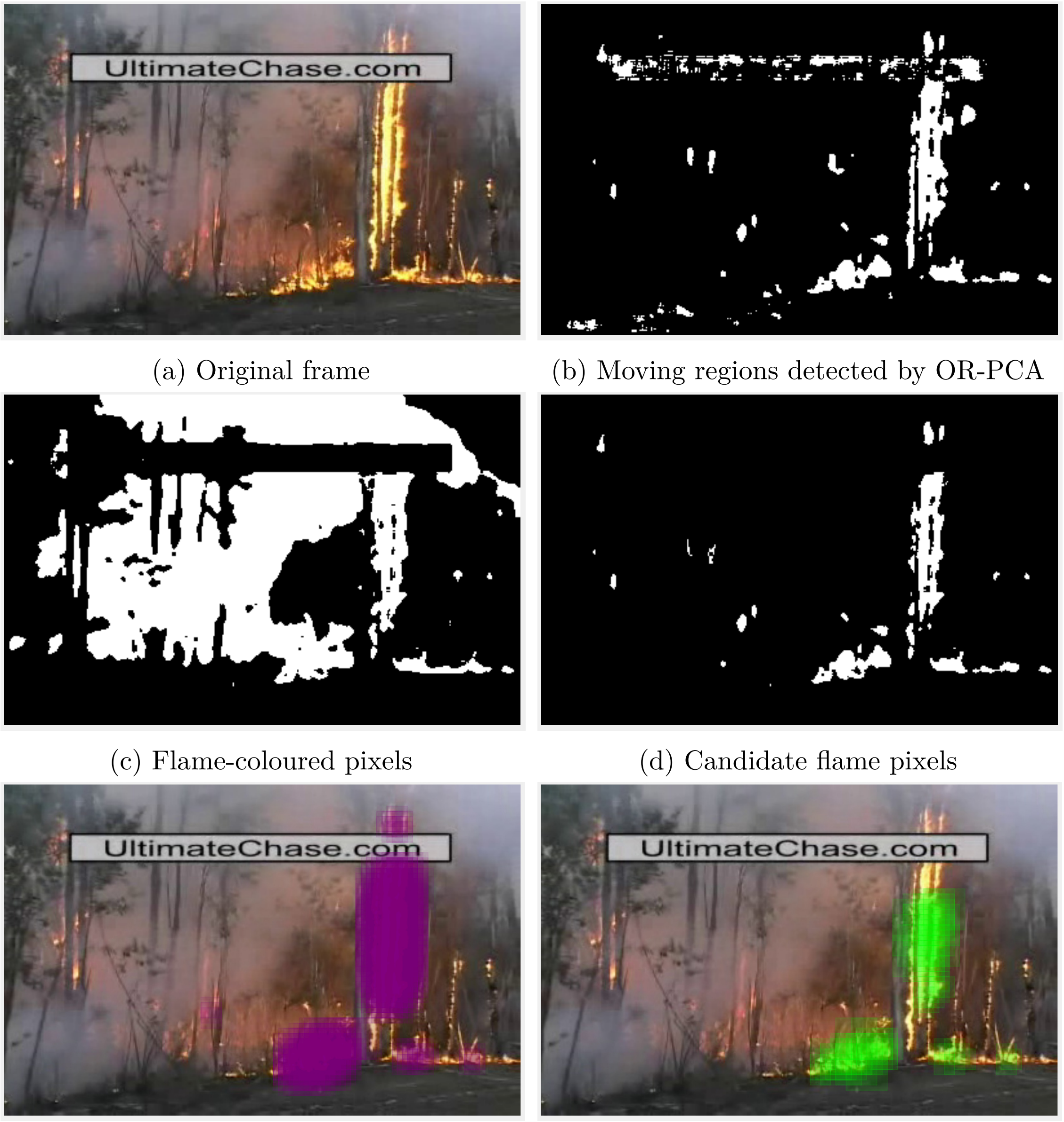
(c) Flame-coloured pixels

(d) Candidate flame pixels

(e) Flame proposals

(f) Detected flame regions

Figure 5.5. True positive results of the framework based on flame R-CNN tested on Video VC7, in which a man walking around burning branches.



(a) Original frame

(b) Moving regions detected by OR-PCA

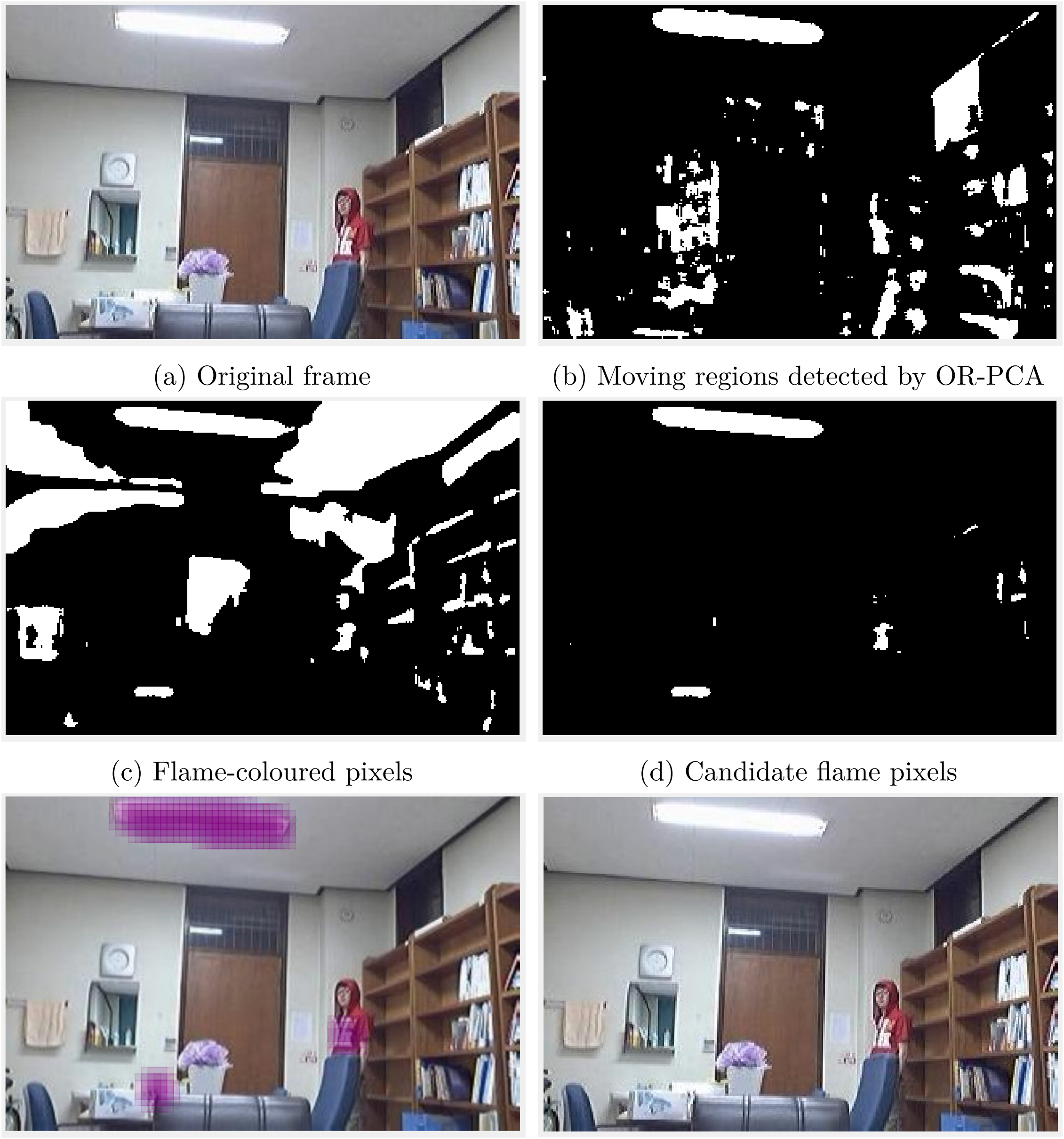
(c) Flame-coloured pixels

(d) Candidate flame pixels

(e) Flame proposals

(f) Detected flame regions

Figure 5.6. True positive results of the flame detection framework based on flame R-CNN tested on Video VC11, in which trees are burning and smoke exists.



(a) Original frame

(b) Moving regions detected by OR-PCA

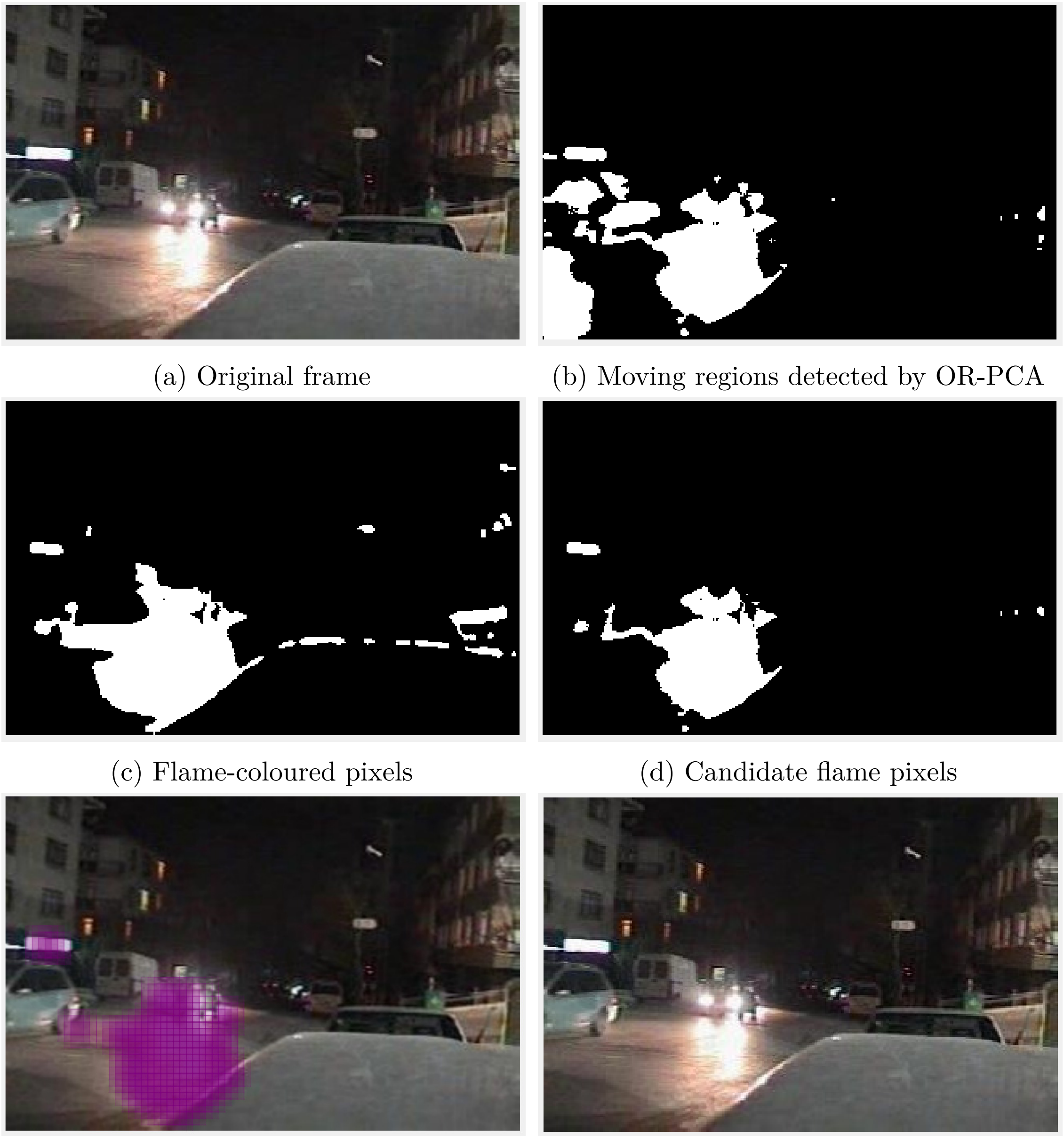
(c) Flame-coloured pixels

(d) Candidate flame pixels

(e) Flame proposals

(f) Detected flame regions

Figure 5.7. True negative results of the flame detection framework based on flame R-CNN tested on Video VC14, in which a man in red walks indoors without flames.


(a) Original frame

(b) Moving regions detected by OR-PCA

(c) Flame-coloured pixels

(d) Candidate flame pixels

(e) Flame proposals

(f) Detected flame regions

Figure 5.8. True negative results of the flame detection framework based on flame R-CNN tested on Video VC12, in which cars move with flashing lights without flames.

Simultaneously, the flame colour model successfully detects the pixels of flame colours. The combination of these two methods detects probable regions of flames based on the dynamic and colour properties, which contributes to the effective generation of flame proposals. The detected candidate flame pixels include the majority (if not all) of flame pixels and a small number of non-flame ones, and thus help to generate proposals which are likely to contain flames in them. For example, the OR-PCA algorithm detects motion caused not only by flames but also by heated airflow and a walking man in Figure 5.5, but most non-flame regions are discarded by the colour model, which helps to reduce the number of proposals without flames, as shown in Figure 5.5e.

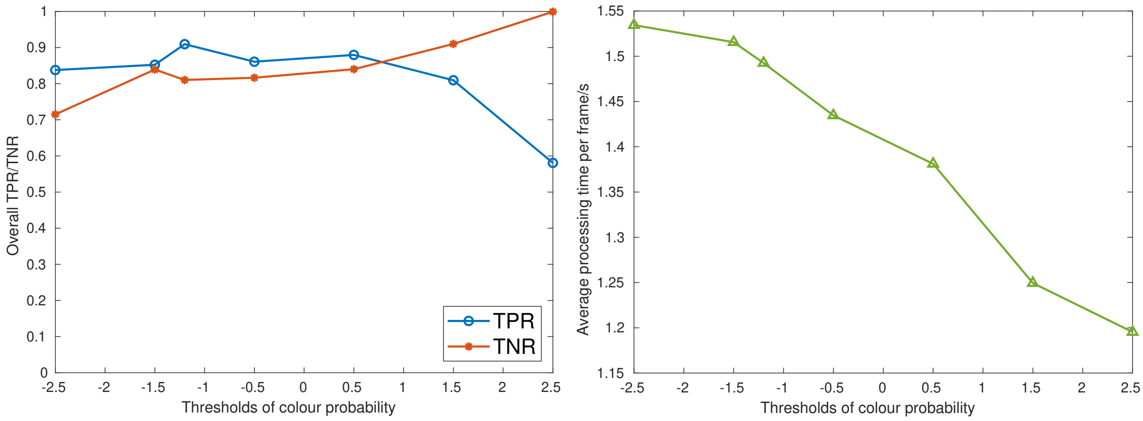
The proposals of flames will be further verified by additional layers of the framework for reliable detection. From the results shown in Figure 5.7 and Figure 5.8, it can be seen that the regions of flashing car lights and the walking person in red clothes are successfully classified as negative by the framework, although they are similar to flames in appearance. Additionally, the proposed framework of flame R-CNN also works well in detecting flame regions, which can be observed from the results shown in Figures 5.5 and 5.6. The video in Figure 5.5 is among the most challenging videos for flame detection, since the semi-transparent colours of the flame regions make the background behind flames visible, resulting in the difficulty in motion detection. Additionally, the texture of the bushes behind flames plays a dominant role when the flames are weak, which mixes the features of the bushes and flames and confuses the trained network. Despite these difficulties, the proposed framework achieves accurate detection of flames in most frames of this video. The non-flame proposals at the left top corner are discarded while the flame regions in the centre are detected successfully. Frame-wise results of all testing videos are given in Table 5.3, with the information of these videos given in Table 3.1.

5.4.3.1 Threshold of the Flame Colour Probability of Flame R-CNN

In the framework of flame R-CNN, a threshold is needed for the flame colour probability in the flame proposal generation scheme, which will influence the performance of detection. In this section, the framework of flame R-CNN is trained and tested using different thresholds to explore their influence on the accuracy and processing time of detection. From Figure 5.9a, it can be seen that the TPR of the framework fluctuates between 0.8 and 0.95 when the threshold increases from -2.5 to 0.5, and has a significant drop with a threshold of 2.5. In contrast, the TNR rises from 0.715 to 0.8392 when the threshold increases from -2.5 to -1.5, and oscillates over the range of thresholds of $[-1.5, 0.5]$. An upward trend in the TNR can be seen with a threshold larger than 1.5. According to the flame proposal generation scheme, a large threshold of colour probability will lead to fewer candidate flame pixels, and

Table 5.3. Detection performance of the framework of flame R-CNN

Videos	tp	fn	tn	fp	Total positive frames	Total negative frames	TPR	TNR
VC1	26	0	0	0	26	0	1	-
VC2	93	0	0	0	93	0	1	-
VC3	48	0	0	0	48	0	1	-
VC4	41	0	0	0	41	0	1	-
VC5	214	9	0	0	214	0	1	-
VC6	176	0	0	0	176	0	1	-
VC7	435	252	4	1	687	5	0.6332	0.8
VC8	572	0	0	69	572	69	1	0
VC9	386	0	0	0	386	0	1	-
VC10	391	4	0	0	395	0	0.9899	-
VC11	186	0	0	0	186	0	1	-
VC12	0	0	130	9	0	139	-	0.9353
VC13	0	0	88	56	0	144	-	0.6111
VC14	0	0	89	66	0	155	-	0.5742
VC15	0	0	364	14	0	378	-	0.9630
VC16	0	0	252	2	0	254	-	0.9921
Overall	2568	256	927	217	2824	1144	0.9093	0.8103



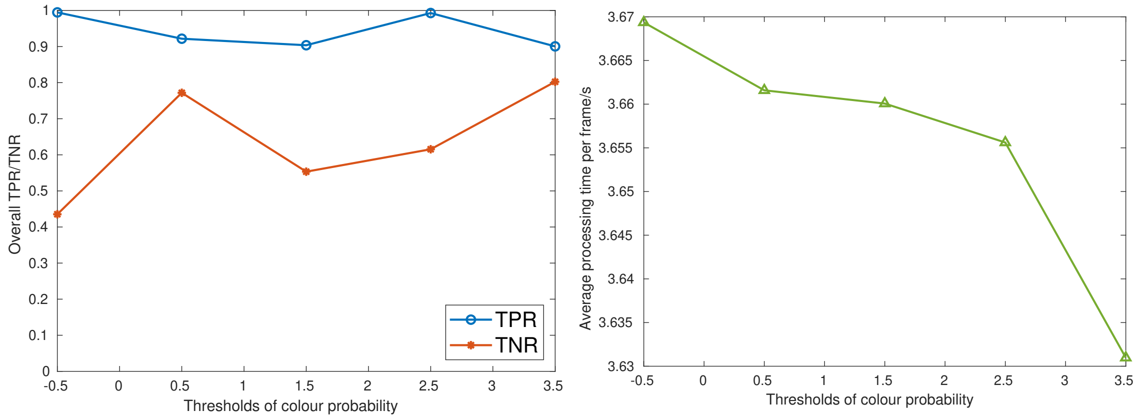
(a) Overall TPRs and TNRs of the flame R-CNN with different thresholds of the estimated colour probability.

(b) Average processing time of one frame of the flame R-CNN with different thresholds of the estimated colour probability.

Figure 5.9. Detection performance and computational complexity of the framework of flame R-CNN with different thresholds of the colour probability.

thus results in a smaller number of flame proposals compared with a small threshold. However, the relationship between the threshold and TPR/ TNR is not monotonic. When the threshold is too large and discards many flame pixels, it will result in a small number of proposals of flames and lead to false negative errors, that explains the sharp decrease in the TPR with when the threshold increases to 2.5. In contrast, a small threshold does not decrease the TNR significantly, since the convolutional features within flame proposals are processed and classified accurately by additional layers. Different from the performance, the computational cost of the framework decreases monotonically with the threshold of flame colour probabilities, which can be seen from Figure 5.9b. It is because a large threshold leads to a small number of proposals that will decrease the computational burden of the framework.

In the flame proposal generation scheme, proposals can also be generated only based on the estimated probability of flame colours. Specifically, candidate pixels of flames are detected only by the colour model, based on which flame proposals are generated (a simplified version of flame R-CNN). The experiments of the simplified flame R-CNN are carried out with different thresholds of flame colour probabilities, of which the TPRs/TNRs and processing time are shown in Figure 5.10. Generally, the TPR and TNR of the simplified flame R-CNN vary with the threshold in a similar way to the full flame R-CNN. The TPR fluctuates with the threshold in the range of 0.5 to 2.5 and suffers a decrease when a large threshold is set. On the contrary, the TNR increases to 0.7719 with a threshold of 0.5 and then swings with an increased threshold. Additionally, the average processing time of one frame declines with thresholds as well, similar to the framework of full flame R-CNN. However, the average processing time per frame is much longer than that of the full flame R-CNN.



(a) Overall TPRs and TNRs of the simplified flame R-CNN with different thresholds of the estimated colour probability.

(b) Average processing time of one frame of the simplified flame R-CNN with different thresholds of the colour probability.

Figure 5.10. Detection performance and computational complexity of the framework of simplified flame R-CNN with different thresholds of the colour probability.

Although the OR-PCA algorithm for motion detection increases the computational cost, the simplified flame R-CNN has a larger number of proposals to process, and thus needs longer time to process each frame compared with the framework of full flame R-CNN given the same threshold. Furthermore, the changes of thresholds also cause larger fluctuations in the performance of the simplified flame R-CNN than the full one, which can be explained by the way of generating flame proposals. In the simplified framework, the threshold of colour probability influences the number of candidate flame pixels as well as proposals directly, while the impact is relieved by the OR-PCA algorithm in the full flame R-CNN.

5.4.4 Comparison of Performance and Discussion

In order to compare the performance of the frameworks of the SqueezeNet, faster R-CNN and flame R-CNN , the TPRs and TNRs of all testing videos are shown in Figure 5.11 and Figure 5.12, respectively. The method of SqueezeNet for flame detection proposed in [2] is fine-tuned with the negative images (without flames) from [62] and the flame images which are used for training the proposed frameworks in this chapter. The numbers of the positive (flame) and negative (no flame) images are roughly the same. The results of the framework of faster R-CNN for flame detection are obtained with four sizes of anchors, of which the details of the parameters are provided in Section 5.4.2. The results of flame R-CNN for comparison are from Table 5.3 of which the threshold of the colour probability is set to -1.2.

The results illustrate that the framework of flame R-CNN achieves higher TPRs than the other two frameworks on all the flame videos except VC7. The framework of faster R-CNN for flame detection successfully detects the flames in most positive videos but fails to provide reliable results in VC6 and VC10. In contrast, the method based on SqueezeNet fails to detect flames in four of the testing videos, with the TPRs of VC6, VC7, VC8 and VC10 lower than 0.2. It is because the SqueezeNet based method conducts classification according to all the features of input frames, resulting in the situation that the features of flames which occupy small regions in the scene are overwhelmed by those features of the background or other salient objects. In contrast, the frameworks of faster R-CNN and flame R-CNN both detect probable regions of flames and provide results only based on the features within those regions, which enhances the accuracy of detecting flames occupying small regions in the scene. However, the sizes of anchors in the framework of faster R-CNN have a large influence on the performance of flame detection. Specifically, the anchor boxes that match the shapes and sizes of flames will lead to good performance, while the flames whose shapes are very different from the anchors can hardly be detected. Since the flames are non-rigid and diverse in shapes, it is difficult to set appropriate sizes of anchors. That is the reason why the framework does not achieve accurate detection in VC6 and VC10, in which the shapes of flames differ significantly with the anchors. The framework based on flame R-CNN does not have this problem since it generates the proposals using the flame proposal generation scheme which is based on a regular grid of boxes and the colour and dynamic properties of flames.

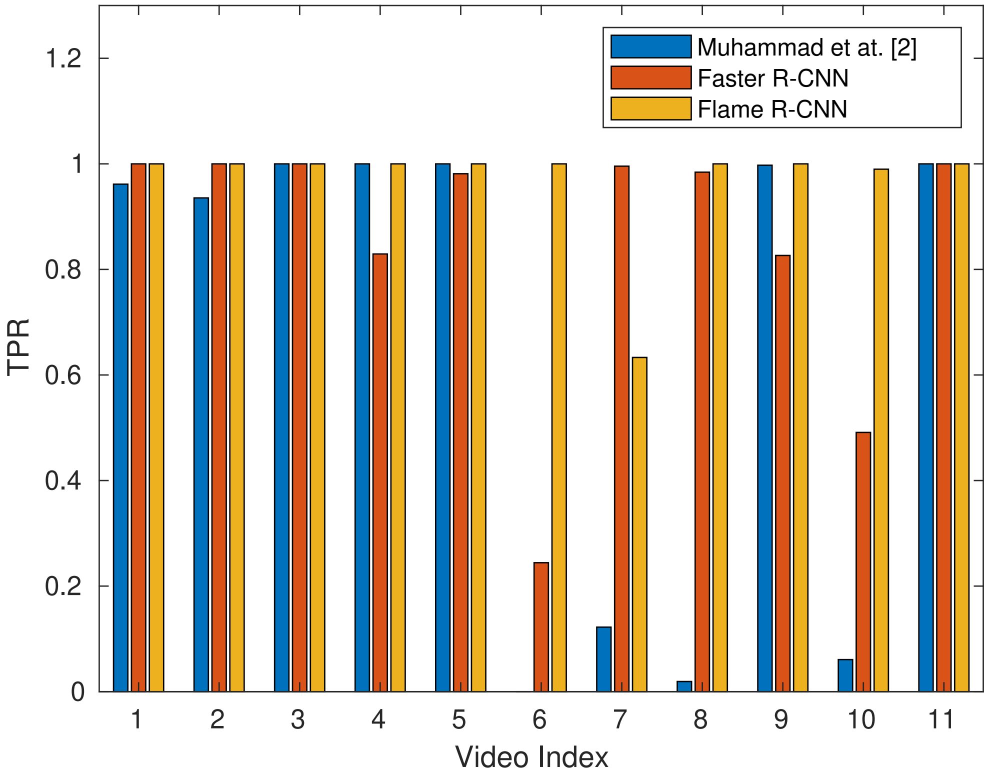


Figure 5.11. TPRs of the method using SqueezeNet proposed by Muhammad et al. [2], and the frameworks of flame R-CNN and faster R-CNN.

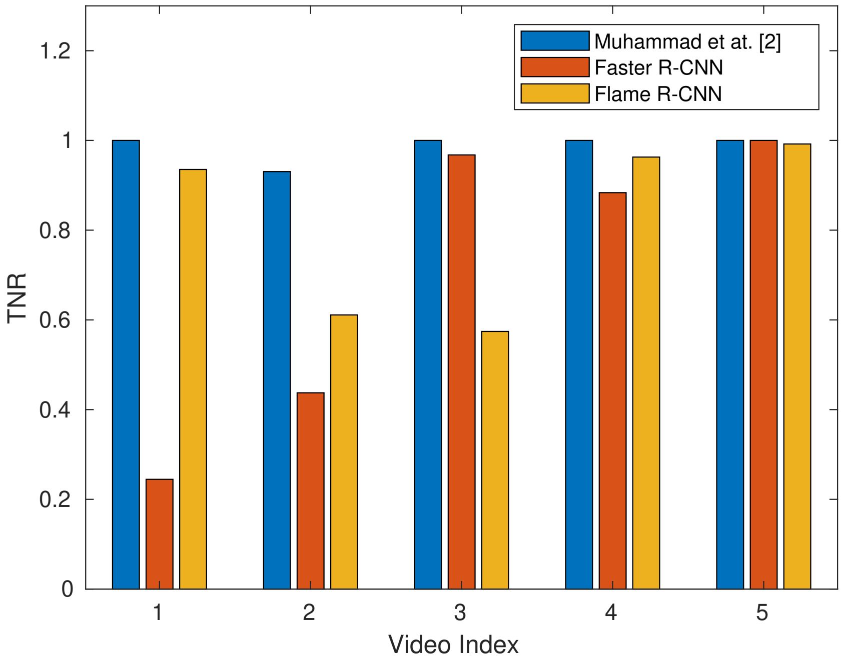


Figure 5.12. TNRs of the method using SqueezeNet proposed by Muhammad et al. [2], and the frameworks of flame R-CNN and faster R-CNN.

On the contrary, the method using SqueezeNet achieves better TNRs than the proposed frameworks. As the TPR and TNR are usually competing, it is not surprising the SqueezeNet has fewer false positive errors than the frameworks of faster R-CNN and flame R-CNN. However, higher TNRs are achieved by the framework of flame R-CNN than faster R-CNN on most negative testing videos, showing the effectiveness of the flame proposal generation scheme.

Generally, the proposed framework of flame R-CNN achieves balanced performance on both positive and negative videos and has higher overall TPR than the other two methods. The framework for flame detection based on faster R-CNN has higher TPRs than the method using SqueezeNet, but lower TNRs. Considering the great losses due to fires every year, the false negative errors of flame detection cause larger damage than the false positive ones and thus should be avoided at all expense. In a nutshell, the framework of flame R-CNN achieves better performance than the methods based on SqueezeNet and faster R-CNN for reducing the losses caused by fires.

5.5 Summary

Two frameworks of flame R-CNN and faster R-CNN are proposed in this chapter, respectively. In the framework of flame R-CNN, a novel flame proposal generation scheme is developed to generate proposals which are likely to contain flames in them. The scheme generates proposals based on two crucial properties of flames, i.e. colours and dynamics. Specifically, candidate flame pixels are detected by the algorithm of OR-PCA and the flame colour model based on the DPGMM. Flame proposals are produced based on a grid of boxes and these candidate pixels of flames. The proposals and a feature map produced by several convolutional layers are combined by an RoI layer to generate a number of small feature maps of a fixed size, which are subsequently processed by additional layers. In the framework of faster R-CNN for flame detection, the proposals are generated by a region proposal network. Similar to the framework of flame R-CNN, the proposals are also projected onto the convolutional feature map to generate features for further processing. Flame regions are outputted by the two frameworks and frame-wise results can be made accordingly.

Since flames have a rich diversity in the appearance, especially in shapes and colours, the flame proposal generation scheme works effectively in generating probable regions of flames by utilizing the dynamic and colour properties of flames, which contributes to high TPRs of the framework of flame R-CNN. Apart from good frame-wise results, the framework of flame R-CNN detects regions of flames accurately as well. In contrast, the performance of the framework based on faster R-CNN is influenced by the sizes of anchor boxes. Appropriately set anchors will lead to accurate detection of flames. However, further research is needed on choosing proper sizes of anchors for flames of diverse shapes.

Chapter 6

Conclusions and Future Work

6.1 Summary and Contributions

This thesis presents research on machine learning based methods for autonomous flame detection in videos. This research area has drawn significantly increasing attention because of its advantages over the conventional techniques for fire detection. Instead of detecting smoke or soot from fires, the video-based approaches for flame detection obtain information from videos and make decisions on the existence of fires using different algorithms and models, which enables the techniques to be easily incorporated into existing surveillance systems. Furthermore, they also achieve faster and more accurate detection of fires compared with the techniques based on the smoke or heat sensors.

Although the promising techniques of video-based flame detection are developing fast, there are still several challenging problems to be solved before widely practical application. The diversity in the appearance of flames increases the difficulty of flame detection. Additionally, suppressing the false alarm rate to an acceptable level is also challenging due to a large amount of interference. Among the challenges, detecting weak or distant flames is one of the most difficult tasks. The semi-transparent colours of weak flames make the objects behind them visible, which mixes the features of flames and the background. Detecting distant flames that are far from surveillance cameras is also challenging since the regions of flames in the scene are small and unclear. Additionally, it is also difficult to distinguish flames from interfering objects, such as flashing car lights or neons, which widely exist in daily life and are similar to flames in both appearance and changing patterns.

To solve the problems discussed above and improve the performance of flame detection, several frameworks based on machine learning methods are proposed in this thesis, which are summarized as follows.

In Chapter 3, a flame detection framework is proposed based on the combination of an optical flow estimation algorithm, a probabilistic saliency analysis scheme, and a temporal wavelet analysis approach. The Horn–Schunck algorithm estimates the optical flow of each pixel by solving an optimisation problem. The probabilistic saliency analysis approach generates two saliency maps based on the intensities and magnitudes of optical flows of pixels, respectively. The two saliency maps are subsequently combined and processed by a group of chromatic selective rules and a temporal wavelet analysis scheme to select probable flame pixels, based on which the final decision of the existence of flames will be made. The framework takes different properties of flames into account to distinguish flame regions from non-flame ones. The overall TPR and TNR of the hybrid framework are both higher than 90%.

In Chapter 4, a novel model of flame colours is developed based on the DPGMM to describe the diverse colours of flames that result from different burning material or various environmental illumination. The distribution of flame colours is modelled by a GMM, of which the prior is set to a DP. As such, the model can learn the number of mixture components from the training data instead of making strong assumptions empirically, which will contribute to the accurate estimation of other parameters. Therefore, the trained model approaches the distribution of flame colours well and can effectively distinguish flame pixels from others based on colours. The inference is accomplished by the algorithms of GS and VI. Experiments show that the proposed colour model outperforms other state-of-the-art models. When incorporated into the framework proposed in Chapter 3, the developed colour model contributes to accurate frame-wise detection of flames in videos.

In Chapter 5, two frameworks are proposed for flame detection based on flame R-CNN and faster R-CNN, respectively. A flame proposal generation scheme is developed in the framework of flame R-CNN to generate proposals which are likely to contain flames in them, based on the OR-PCA algorithm and the flame colour model introduced in Chapter 4. The proposals are subsequently projected onto a feature map produced by several convolutional layers to generate a number of small feature maps of a fixed size, which will be processed by additional layers and output detected regions of flames. In the framework of faster R-CNN, proposals of flames are generated by a region proposal network instead of the flame proposal generation scheme. The generated proposals are processed in a similar way to the flame R-CNN.

The results of experiments show that the framework of flame R-CNN accurately detects regions of flames, and achieves a good performance of frame-wise detection. The performance of the framework has proven the effectiveness of the flame proposal generation scheme, which generates proposals based on the dynamic and colour properties of flames. In contrast, the region proposal network in the framework based on faster R-CNN is sensitive to the sizes of anchor boxes. It can accurately detect the flames of which the sizes match the anchors, while fails to detect the flames that differ significantly from the anchor boxes in shape. The diversity in the appearance of flames makes it difficult to set anchors which can match all flames. Therefore, the flame proposal generation scheme works more effectively in generating proposals than the region proposal network and thus contributes to the good performance of the framework of flame R-CNN.

6.2 Future Work

In this section, some promising approaches which may improve the current methods of flame detection will be discussed, together with the potential applications of the proposed frameworks to other research areas.

- **The framework of flame R-CNN can be employed in the area of non-rigid object detection**. Different from the rigid targets in most detection tasks, non-rigid objects usually have more diverse appearance, especially shapes. This property makes it challenging for most existing frameworks (e.g. fast R-CNN [93] or YOLO [96]) to detect non-rigid targets accurately. In contrast, the proposed framework of flame R-CNN generates region proposals based on the key properties of the targets (e.g., colour and dynamics in flame detection), which is expected to work better in the detection of non-rigid objects.

- **The DPGMM can be used to model the distribution of other features,** especially those whose distributions are not well known or difficult to model with typical distributions. Theoretically, any distribution can be estimated accurately by a GMM given a proper number of mixture components, which is not known to researchers in most cases. This problem can be solved by the proposed framework to learn the parameters from data instead of making strong assumptions. As such, accurate statistical models of features can be obtained.

- **Incorporating the feature pyramid network (FPN) [103] into the current framework of flame R-CNN to achieve multi-scale detection**. The scales of flame regions can be significantly different because of the various intensities of combustion and different distances between fires and cameras, so a multi-scale system is expected to considerably improve the performance of detection. The FPN utilizes the inherent pyramidal hierarchy of CNNs and can significantly decrease the computational complexity of pyramid representation, which is crucial in video processing tasks.

- **Performing super-resolution on each frame of testing videos to improve the detection results.** Most frames of videos from surveillance systems are of low resolution which is limited by the costs of facilities. The videos of low quality lose important information of details which can help to distinguish flames from other objects. For example, the size of most testing videos are 240 × 320 which is not as big as the inputs of most popular CNNs, and the region proposals are even smaller. The Gaussian process [104] based single image super-resolution method [105] does not need to train with any external data in advance. Instead, it performs both training and prediction based on the low-resolution frames, which makes the framework more transferable to diverse videos.

- **Improving the architecture of layers to emphasize the features unique to flames.** As the proposed flame R-CNN framework employs a grid of boxes to generate flame proposals, the texture related features play a more crucial role than shapes in flame detection. Therefore, revising the architecture of layers to emphasize the texture is expected to enhance the detection performance [106]. Further improvement may be achieved by learning the architectures of CNNs automatically with the neural architecture search [107], which outperforms many manually designed state-of-the-art deep neural networks.

Bibliography

[1] F. Li, R. Krishna, and D. Xu, "CS231n convolutional neural networks for visual recognition," http://cs231n.github.io/convolutional-networks/.

[2] K. Muhammad, J. Ahmad, Z. Lv, P. Bellavista, P. Yang, and S. W. Baik, "Efficient deep CNN-based fire detection and localization in video surveillance applications," *IEEE Trans. Syst., Man, and Cybern.: Syst.*, vol. 49, no. 7, pp. 1419–1434, 2018.

[3] T. Chen, P. Wu, and Y. Chiou, "An early fire-detection method based on image process." in *Proc. Int. Conf. Image Process.*, Singapore, 2004, pp. 1707–1710.

[4] T. Celik and H. Demirel, "Fire detection in video sequences using a generic color model," *Fire Safety J.*, vol. 44, no. 2, pp. 147–158, 2009.

[5] B. U. Töreyin, Y. Dedeoğlu, U. Güdükbay, and A. E. Cetin, "Computer vision based method for real-time fire and flame detection," *Pattern Recognition Lett.*, vol. 27, no. 1, pp. 49–58, 2006.

[6] J. R. Hall, "The total cost of fire in the United States," http://www.nfpa.org/news-and-research/fire-statistics-and-reports/fire-statistics/fires-in-the-us/overall-fire-problem/total-cost-of-fire/.

[7] E. Crowhurst, "Fire statistics monitor: April 2014 to March 2015," https://www.gov.uk/government/statistics/.

[8] "Official website of the department of homeland security of America," https://www.ready.gov/home-fires/.

[9] M. Bugarić, T. Jakovčević, and D. Stipaničev, "Adaptive estimation of visual smoke detection parameters based on spatial data and fire risk index," *Comput. Vision and Image Understanding*, vol. 118, pp. 184–196, 2014.

[10] A. E. Çetin, K. Dimitropoulos, B. Gouverneur, N. Grammalidis, O. Günay, Y. H. Habiboğlu, and et al., "Video fire detection–review," *Digital Signal Process.*, vol. 23, no. 6, pp. 1827–1843, 2013.

[11] F. Yuan, "A fast accumulative motion orientation model based on integral image for video smoke detection," *Pattern Recognition Lett.*, vol. 29, no. 7, pp. 925–932, 2008.

[12] P. Borges and E. Izquierdo, "A probabilistic approach for vision-based fire detection in videos," *IEEE Trans. Circuits and Syst. for Video Technol.*, vol. 20, no. 5, pp. 721–731, 2010.

[13] C. Liu and N. Ahuja, "Vision based fire detection," in *Proc. 17th Int. Conf. Pattern Recognition*, Cambridge, UK, 2004, pp. 134–137.

[14] Z. Li, L. S. Mihaylova, O. Isupova, and L. Rossi, "Autonomous flame detection in videos with a Dirichlet process Gaussian mixture color model," *IEEE Trans. Ind. Informat.*, vol. 14, no. 3, pp. 1146–1154, 2017.

[15] B. K. Horn and B. G. Schunck, "Determining optical flow," *Artificial Intell.*, vol. 17, pp. 185–203, 1981.

[16] Z. Li, O. Isupova, L. Mihaylova, and L. Rossi, "Autonomous flame detection in video based on saliency analysis and optical flow," in *Proc. IEEE Int. Conf. Multisensor Fusion and Integration for Intelligent Syst.*, Baden-Baden, Germany, 2016, pp. 218–223.

[17] T. Celik, H. Demirel, H. Ozkaramanli, and M. Uyguroglu, "Fire detection using statistical color model in video sequences," *J. of Visual Commun. and Image Representation*, vol. 18, no. 2, pp. 176–185, 2007.

[18] H. Yamagishi and J. Yamaguchi, "A contour fluctuation data process. method for fire flame detection using a color camera," in *Proc. 26th Annu. Conf. of Ind. Electron. Soc.*, Nagoya, Japan, 2000, pp. 824–829.

[19] Y. H. Habiboğlu, O. Günay, and A. E. Çetin, "Covariance matrix-based fire and flame detection method in video," *Mach. Vision and Appl.*, vol. 23, no. 6, pp. 1103–1113, 2012.

[20] D. Y. Chino, L. P. Avalhais, J. F. Rodrigues, and A. J. Traina, "Bowfire: detection of fire in still images by integrating pixel color and texture analysis," in *Proc. 28th Conf. Graph., Patterns and Images*, Salvador, Brazil, 2015, pp. 95–102.

[21] P. Foggia, A. Saggese, and M. Vento, "Real-time fire detection for video-surveillance applications using a combination of experts based on color, shape, and motion," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 9, pp. 1545–1556, 2015.

[22] M. Mueller, P. Karasev, I. Kolesov, and A. Tannenbaum, “Optical flow estimation for flame detection in videos,” *IEEE Trans. Image Process.*, vol. 22, no. 7, pp. 2786–2797, 2013.

[23] R. Chi, Z. Lu, and Q. Ji, “Real-time multi-feature based fire flame detection in video,” *IET Image Process.*, vol. 11, no. 1, pp. 31–37, 2016.

[24] T. Celik, H. Demirel, and H. Ozkaramanli, “Automatic fire detection in video sequences,” in *Proc. 14th Eur. Signal Process. Conf.*, Florence, Italy, 2006, pp. 1–5.

[25] J. Chen, Y. He, and J. Wang, “Multi-feature fusion based fast video flame detection,” *Building and Environment*, vol. 45, no. 5, pp. 1113–1122, 2010.

[26] C. Stauffer and W. E. L. Grimson, “Adaptive background mixture models for real-time tracking,” in *Proc. Comput. Soc. Conf. Comput. Vision and Pattern Recognition*, vol. 2, Fort Collins, CO, USA, 1999.

[27] D. Han and B. Lee, “Flame and smoke detection method for early real-time detection of a tunnel fire,” *Fire Safety J.*, vol. 44, no. 7, pp. 951–961, 2009.

[28] D. Wang, X. Cui, E. Park, C. Jin, and H. Kim, “Adaptive flame detection using randomness testing and robust features,” *Fire Safety J.*, vol. 55, pp. 116–125, 2013.

[29] V. B. Celen and M. F. Demirci, “Fire detection in different color models,” in *Proc. Int. Conf. Image Process., Comput. Vision, and Pattern Recognition*, Las Vegas, NV, USA, 2012, pp. 1–7.

[30] T. Toulouse, L. Rossi, T. Celik, and M. Akhloufi, “Automatic fire pixel detection using image process.: a comparative analysis of rule-based and machine learning-based methods,” *Signal, Image and Video Process.*, vol. 10, no. 4, pp. 647–654, 2016.

[31] B. C. Ko, K. Cheong, and J. Nam, “Fire detection based on vision sensor and support vector machines,” *Fire Safety J.*, vol. 44, no. 3, pp. 322–329, 2009.

[32] M. Kandil and M. Salama, “A new hybrid algorithm for fire vision recognition,” in *Proc. IEEE EUROCON*, St.Petersburg, Russia, 2009, pp. 1460–1466.

[33] B. C. Ko, S. J. Ham, and J. Y. Nam, “Modeling and formalization of fuzzy finite automata for detection of irregular fire flames,” *IEEE Trans. Circuits and Syst. for Video Technol.*, vol. 21, no. 12, pp. 1903–1912, 2011.

[34] F. Yuan, “A double mapping framework for extraction of shape-invariant features based on multi-scale partitions with AdaBoost for video smoke detection,” *Pattern Recognition*, vol. 45, no. 12, pp. 4326–4336, 2012.

[35] S. G. Kong, D. Jin, S. Li, and H. Kim, “Fast fire flame detection in surveillance video using logistic regression and temporal smoothing,” *Fire Safety J.*, vol. 79, pp. 37–43, 2016.

[36] B. Ko, K. Cheong, and J. Nam, “Early fire detection algorithm based on irregular patterns of flames and hierarchical Bayesian networks,” *Fire Safety J.*, vol. 45, no. 4, pp. 262–270, 2010.

[37] O. Gunay, B. U. Toreyin, K. Kose, and A. E. Cetin, “Entropy-functional-based online adaptive decision fusion framework with application to wildfire detection in video,” *IEEE Trans. Image Process.*, vol. 21, no. 5, pp. 2853–2865, 2012.

[38] A. Namozov and Y. Cho, “An efficient deep learning algorithm for fire and smoke detection with limited data,” *Advances in Elect. and Comput. Eng.*, vol. 18, no. 4, pp. 121–129, 2018.

[39] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, “Squeezenet: AlexNet-level accuracy with 50x fewer parameters and $<$ 0.5 MB model size,” *arXiv preprint arXiv:1602.07360*, 2016.

[40] L. Merino, F. Caballero, J. R. Dios, J. Ferruz, and A. Ollero, “A cooperative perception system for multiple UAVs: Application to automatic detection of forest fires,” *J. of Field Robotics*, vol. 23, no. 3-4, pp. 165–184, 2006.

[41] L. Merino, F. Caballero, J. R. Dios, I. Maza, and A. Ollero, “An unmanned aircraft system for automatic forest fire monitoring and measurement,” *J. of Intelligent & Robotic Syst.*, vol. 65, no. 1-4, pp. 533–548, 2012.

[42] W. Schroeder, P. Oliva, L. Giglio, B. Quayle, E. Lorenz, and F. Morelli, “Active fire detection using Landsat-8/OLI data,” *Remote Sensing of Environment*, vol. 185, pp. 210–220, 2016.

[43] T. Bouwmans, “Traditional and recent approaches in background modeling for foreground detection: An overview,” *Comput. science review*, vol. 11, pp. 31–66, 2014.

[44] M. Piccardi, “Background subtraction techniques: a review,” in *Proc. IEEE Int. Conf. Syst., Man and Cybern.*, vol. 4, The Hague, Netherlands, 2004, pp. 3099–3104.

[45] M. Yu, A. Rhuma, S. M. Naqvi, L. Wang, and J. Chambers, "A posture recognition-based fall detection system for monitoring an elderly person in a smart home environment," *IEEE Trans. Inform. Technol. in Biomedicine*, vol. 16, no. 6, pp. 1274–1286, 2012.

[46] B. U. Töreyin, "Fire detection algorithms using multimodal signal and image analysis," Ph.D. dissertation, Bilkent University, 2009.

[47] K. P. Murphy, *Machine Learning: a Probabilistic Perspective.* MIT press, 2012.

[48] C. M. Bishop, *Pattern Recognition and Machine Learning.* Springer, 2006.

[49] L. Martino, J. Read, and D. Luengo, "Independent doubly adaptive rejection Metropolis sampling within Gibbs sampling," *IEEE Trans. Signal Process.*, vol. 63, no. 12, pp. 3123–3138, 2015.

[50] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational inference: A review for statisticians," *J. of the Amer. Statistical Assoc.*, vol. 112, no. 518, pp. 859–877, 2017.

[51] J. M. Joyce, "Kullback-leibler divergence," in *Int. Encyclopedia of Statistical Science.* Springer, 2011, pp. 720–722.

[52] M. D. Hoffman, D. M. Blei, C. Wang, and J. Paisley, "Stochastic variational inference," *The J. of Mach. Learning Research*, vol. 14, no. 1, pp. 1303–1347, 2013.

[53] K. Kurihara, M. Welling, and N. Vlassis, "Accelerated variational dirichlet process mixtures," in *Proc. Advances in Neural Inform. Process. Syst.*, Vancouver, BC, Canada, 2006, pp. 761–768.

[54] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, 2015.

[55] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, "Deep learning for computer vision: A brief review," *Computational Intell. and Neuroscience*, vol. 2018, pp. 1–13, 2018.

[56] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning.* MIT press, 2016.

[57] Y.-L. Boureau, J. Ponce, and Y. LeCun, "A theoretical analysis of feature pooling in visual recognition," in *Proc. 27th Int. Conf. Machine Learning*, Haifa, Israel, 2010, pp. 111–118.

[58] X. Glorot, A. Bordes, and Y. Bengio, “Deep sparse rectifier neural networks,” in *Proc. 14th Int. Conf. artificial Intell. and statistics*, Fort Lauderdale, FL, USA, 2011, pp. 315–323.

[59] L. Pauly, D. Hogg, R. Fuentes, and H. Peel, “Deeper networks for pavement crack detection,” in *Proc. 34th Int. Symp. in Automation and Robotics in Construction*, Taipei, China, 2017, pp. 479–485.

[60] K. He, X. Zhang, S. Ren, and J. Sun, “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification,” in *Proc. IEEE Int. Conf. computer vision*, Santiago, Chile, 2015, pp. 1026–1034.

[61] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[62] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Proc. Advances in Neural Inform. Process. Syst.*, Lake Tahoe, CA, USA, 2012, pp. 1097–1105.

[63] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov *et al.*, “Going deeper with convolutions,” in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition*, Boston, MA, USA, 2015, pp. 1–9.

[64] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the inception architecture for computer vision,” in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition*, Las Vegas, NV, USA, 2016, pp. 2818–2826.

[65] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, “Inception-v4, inception-resnet and the impact of residual connections on learning,” in *Proc. Thirty-First AAAI Conf. Artificial Intell.*, San Francisco, CA, USA, 2017.

[66] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.

[67] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition*, Las Vegas, NV, USA, 2016, pp. 770–778.

[68] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and L. FeiFei, “Imagenet: A large-scale hierarchical image database,” in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition*, Miami, FL, USA, 2009, pp. 248–255.

[69] Y. Hsiao, C. Chuang, J. Jiang, and C. Chien, “A contour based image segmentation algorithm using morphological edge detection,” in *Proc. IEEE Int. Conf. Syst., Man and Cybern.*, Waikoloa, HI, USA, 2005, pp. 2962–2967.

[70] C. Fox, *An Introduction To the Calculus of Variations.* Courier Corporation, 1987.

[71] Y. Jia, J. Yuan, J. Wang, J. Fang, Q. Zhang, and Y. Zhang, “A saliency-based method for early smoke detection in video sequences,” *Fire Technol.*, pp. 1–22, 2015.

[72] E. Rahtu and J. Heikkilä, “A simple and efficient saliency detector for background subtraction,” in *Proc. 12th Int. Conf. Comput. Vision Workshops*, Kyoto, Japan, 2009, pp. 1137–1144.

[73] “Computer vision and pattern recognition laboratory, Keimyung University, Korea,” http://cvpr.kmu.ac.kr/.

[74] A. E. Cetin, “Computer vision based fire detection software,” http://signal.ee.bilkent.edu.tr/VisiFire/.

[75] T. Fawcett, “An introduction to ROC analysis,” *Pattern Recognition Lett.*, vol. 27, no. 8, pp. 861–874, 2006.

[76] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei, “Hierarchical Dirichlet processes,” *J. Amer. Statistical Assoc.*, vol. 101, no. 476, pp. 1566–1581, 2006.

[77] V. I. Bogachev, *Measure Theory.* Springer, 2007, vol. 1.

[78] S. J. Gershman and D. M. Blei, “A tutorial on Bayesian nonparametric models,” *J. Math. Psychology*, vol. 56, no. 1, pp. 1–12, 2012.

[79] R. M. Neal, “Markov chain sampling methods for Dirichlet process mixture models,” *J. Computational and Graphical Statist.*, vol. 9, no. 2, pp. 249–265, 2000.

[80] D. M. Blei and M. I. Jordan, “Variational inference for Dirichlet process mixtures,” *Bayesian Analysis*, vol. 1, no. 1, pp. 121–143, 2006.

[81] B. Efron, “The geometry of exponential families,” *The Annals of Statist.*, vol. 6, no. 2, pp. 362–376, 1978.

[82] J. M. Bernardo, “Algorithm AS 103: Psi (digamma) function,” *J. of the Roy. Statistical Soc. Series C (App. Statist.)*, vol. 25, no. 3, pp. 315–317, 1976.

[83] J. L. Bentley, “Multidimensional binary search trees used for associative searching,” *Commun. of the ACM*, vol. 18, no. 9, pp. 509–517, 1975.

[84] T. Toulouse, L. Rossi, M. Akhloufi, T. Celik, and X. Maldague, “Benchmarking of wildland fire colour segmentation algorithms,” *Image Process.*, vol. 9, no. 12, pp. 1064–1072, 2015.

[85] K. P. Murphy, “Conjugate Bayesian analysis of the Gaussian distribution,” Tech. Rep., 2007.

[86] I. Jolliffe, *Principal Component Analysis.* Elsevier, 2011.

[87] E. J. Candès, X. Li, Y. Ma, and J. Wright, “Robust principal component analysis?” *J. of the ACM*, vol. 58, no. 3, pp. 1–37, 2011.

[88] B. Recht, M. Fazel, and P. A. Parrilo, “Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization,” *SIAM Review*, vol. 52, no. 3, pp. 471–501, 2010.

[89] T. Bouwmans and E. H. Zahzah, “Robust PCA via principal component pursuit: A review for a comparative evaluation in video surveillance,” *Comput. Vision and Image Understanding*, vol. 122, pp. 22–34, 2014.

[90] J. Feng, H. Xu, and S. Yan, “Online robust PCA via stochastic optimization,” in *Proc. Advances in Neural Inform. Process. Syst.*, Lake Tahoe, CA, USA, 2013, pp. 404–412.

[91] J. Pang, K. Chen, J. Shi, H. Feng, W. Ouyang, and D. Lin, “Libra R-CNN: Towards balanced learning for object detection,” in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition*, Long Beach, CA, USA, 2019, pp. 821–830.

[92] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition*, Columbus, OH, USA, 2014, pp. 580–587.

[93] R. Girshick, “Fast R-CNN,” in *Proc. IEEE Int. Conf. Computer Vision*, Las Condes, Chile, 2015, pp. 1440–1448.

[94] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” in *Proc. Advances in Neural Inform. Process. Syst.*, Montreal, Canada, 2015, pp. 91–99.

[95] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask R-CNN,” in *Proc. IEEE Int. Conf. Computer Vision*, Venice, Italy, 2017, pp. 2961–2969.

[96] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition*, Las Vegas, NV, USA, 2016, pp. 779–788.

[97] J. Redmon and A. Farhadi, “YOLO9000: better, faster, stronger,” in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition*, Honolulu, HA, USA, 2017, pp. 7263–7271.

[98] ——, “YOLOv3: An incremental improvement,” *arXiv*, 2018.

[99] J. R. Uijlings, K. E. Sande, T. Gevers, and A. W. Smeulders, “Selective search for object recognition,” *Int. J. of Comput. Vision*, vol. 104, no. 2, pp. 154–171, 2013.

[100] R. M. Haralick and L. G. Shapiro, *Computer and Robot Vision.* Addison-wesley Reading, 1992, vol. 1.

[101] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma *et al.*, “Imagenet large scale visual recognition challenge,” *Int. J. of Comput. Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[102] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.

[103] T. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, “Feature pyramid networks for object detection,” in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition*, Honolulu, HA, USA, 2017, pp. 2117–2125.

[104] C. E. Rasmussen, “Gaussian processes in machine learning,” in *Proc. Summer School on Mach. Learning*, 2003, pp. 63–71.

[105] H. He and W.-C. Siu, “Single image super-resolution using Gaussian process regression,” in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition*, Colorado Springs, CL, USA, 2011, pp. 449–456.

[106] J. Xue, H. Zhang, and K. Dana, “Deep texture manifold for ground terrain recognition,” in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition*, Salt Lake City, UT, USA, 2018, pp. 558–567.

[107] G. Ghiasi, T. Lin, and Q. V. Le, “NAS-FPN: Learning scalable feature pyramid architecture for object detection,” in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition*, Long Beach, CA, USA, 2019, pp. 7036–7045.