Republic of Iraq

Ministry of Higher Education & Scientific Research

University of Kerbala

College of Engineering

Electrical and Electronic Engineering Department

An Automatic System for Fire Detection and Localization based on Image Processing Techniques

A Thesis Submitted to the Council of the Faculty of the College of the Engineering/University Of Kerbala in Partial Fulfillment of the Requirements for the Master Degree in Electrical and Electronic Engineering

By:

Zahraa Shihab Ahmed Ali Azghar

Supervisors

Prof. Dr. Haider Ismael Shahadi

Prof. Dr. Hawraa Hasan Abass

June 2023 Dhu al-Qa’dah 1444

Republic of Iraq

Ministry of Higher Education & Scientific Research

University of Kerbala

College of Engineering

Electrical and Electronic Engineering Department

An Automatic System for Fire Detection and Localization based on Image Processing Techniques

A Thesis Submitted to the Council of the Faculty of the College of the Engineering/University Of Kerbala in Partial Fulfillment of the Requirements for the Master Degree in Electrical and Electronic Engineering

By:

Zahraa Shihab Ahmed Ali Azghar

Supervisors

Prof. Dr. Haider Ismael Shahadi

Prof. Dr. Hawraa Hasan Abass

June 2023 Dhu al-Qa’dah 1444

بسم الله الرحمن الرحيم

يَرْفَعِ اللَّهُ الَّذِينَ آمَنُوا مِنكُمْ وَالَّذِينَ أُوتُوا

الْعِلْمَ دَرَجَاتٍ

صدق الله العلي العظيم

(المجادلة: من الآية 11)

Examination committee certification

We certify that we have read the thesis entitled "**An Automatic System for Fire Detection and Localization based on Image Processing Techniques,**" and as an examining committee, we examined the student " **Zahraa Shihab Ahmed Ali Azghar** " in its content and in what is connected with it and that, in our opinion, it is adequate as a thesis for the degree of Master of Science in Electrical and Electronic Engineering.

Chairman
Signature:
Prof. Dr. Kasim Karam Abdulla
Date:26/6/2023

Member
Signature: Farhan
Assist. Prof. Dr. Hameed R. Farhan
Date:22/6/2023

Member
Signature:
Dr. Arwa Hameed Mohammed Taqi
Date:25/6/2023

Supervisor
Signature:
Prof. Dr. Haider Ismael Shahadi
Date: / / 2023

Supervisor
Signature:
Prof. Dr. Hawraa Hasan Abass
Date:25/6/2023

Signature:
Prof. Dr. Haider Ismael Shahadi
Head of the Department of Electrical and Electronic Engineering
Date:16/7/2023

Signature: L.ShRas
Prof. Dr. Laith Sh. Rasheed
Dean of the Engineering College
Date: / / 2023

Supervisor certificate

We certify that the thesis entitled **"An Automatic System for Fire Detection and Localization based on Image Processing Techniques"** was prepared by **Zahraa Shihab Ahmed Ali Azghar** under our supervision at the Department of Electrical and Electronic Engineering, Faculty of Engineering, University of Kerbala as a partial of fulfilment of the requirements for the Degree of Master of Science in Electrical and Electronic Engineering.

Signature:
Prof. Dr. Haider Ismael Shahadi
Date:25/6/ 2023

Signature:
Prof. Dr. Hawraa Hasan Abass
Date25/ 6/ 2023

Linguistic certificate

I certify that the thesis entitled "**An Automatic System for Fire Detection and Localization based on Image Processing Techniques,**" which has been submitted by **Zahraa Shihab Ahmed Ali Azghar,** has been proofread, and its language has been amended to meet the English style.

Signature:
Dr. Hussein Al-Hamdani
Date: 25/ 6 / 2023

Undertaking

I certify that research work titled "**An Automatic System for Fire Detection and Localization based on Image Processing Techniques**" is my own work. The work has not been presented elsewhere for assessment. Where material has been used from other sources, it has been properly acknowledged / referred.

Signature:

Zahraa Shihab Ahmed Ali Azghar

Date: 16/7 / 2023

Dedication

To my beloved father, who passed away before he could see me complete this work. Your unwavering support, encouragement, and love will always remain with me. You have been a constant source of inspiration, and I dedicate this work to your memory.

To my dear mother, who has been my rock throughout my life. Your love, patience, and unwavering support have been the foundation of my success. I am grateful for your sacrifices and guidance, and I dedicate this work to you.

To my loving husband and children, who have been my pillars of strength and my reason for living. Your love, understanding, and patience have helped me navigate the ups and downs of life. I dedicate this work to you, with all my heart.

To my supervisor, who has been an invaluable mentor, and guide. Your wisdom, expertise, and support have been instrumental in shaping this project. I am grateful for your belief in me and your unwavering encouragement. I dedicate this work to you with sincere thanks.

Acknowledgments

In the name of Allah, Most Gracious, Most Merciful

First of all, praise is to Allah, the lord of the whole creation, for all the blessings were the help in achieving this research to its end.

I wish to express my thanks to my supervisors, Prof. Dr. Haider Ismael Shahadi and Prof. Dr. Hawraa Hasan Abass, for supervising this research and for their generosity, patience, and continuous guidance throughout the work. It has been my good fortune to have their advice and guidance.

I would also like to thank the Department of Electrical and Electronic Engineering, in particular, and the College of Engineering, in general, for helping us to complete the study.

I offer my thanks to all people who have stood beside me during my studying in Karbala University, but I cannot mention their names here one by one.

I would like to express my gratitude and great love to my husband, mother, children, family, and friends, who were unlimited support and patience me.

Abstract

The rise of climate temperatures has led to increase fires in buildings, farms, and wheat fields. Fire detection-based smoke sensor is unsuitable for open and large buildings, and outdoor areas. This thesis proposes a fire detection and localization system based on a suitable video camera for indoor and outdoor areas that can withstand environmental changes such as illuminance and color correlation to fire flames and smoke.

The proposed fire detection system combines color and motion detection approaches for accurate results. It uses *RGB* video input, which is decomposed using wavelet transform to reduce data processing while preserving fire features. The decomposed frames are then processing for color and motion detection to identify fire characteristics. Morphological post-processing removes unwanted pixels, and the detected fire area is calculated and bounded if it meets the threshold conditions. The system employs three threshold types, including static multi-threshold, non-adaptive threshold for different color spaces, and adaptive threshold using optimization algorithms based on objective functions like Otsu and Kapur.

The fire localization is achieved through the inverse camera parameters calibration and position of fire in pixels. The approach uses the projective transformation matrix that is mapped between frame pixels and real-world positions.

MATLAB R2021b used to implement the proposed fire detection system. The experimental results of the proposed system are achieved offline and online. The offline tests utilize the datasets, including KMU, VisiFire, and FireSense, as well as our recorded videos. The downloaded datasets include 92 fire videos, both smoke- and flame-based. The offline test achieved about

92.1% average rate of the correct fire detection for the static multi-threshold smoke detection system. In contrast, the flame detection system-based non-adaptive threshold achieved about 94%. Moreover, the adaptive thresholds flame detection system achieved approximately 96%.

Online fire detection is tested in indoor and outdoor places. The indoor average accuracy was 90% for the static multi-threshold smoke detection system and 97.5% for the adaptive threshold flame detection system. The average outdoor fire detection accuracy for the static multi-threshold smoke detection system, non-adaptive, and adaptive thresholds flame detection systems was 94.1%, 94.4%, and 92.4%, respectively. The error of fire localization is achieved at less than 0.42 meters.

The proposed system can be successfully used for fire detection in real-time with high accuracy, whether indoors or outdoors, and for different environmental conditions.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

2D	Two Dimension
3D	Three Dimension
ABC	Artificial Bee Colony
ACC	Accuracy
BSM	Background Subtraction Method
CCTV	Closed-Circuit Television
CIE L*a*b*	International Commission on Illumination, Luminance, Four unique colors
CNN	Convolutional Neural Networks
CWT	Continue Wavelet Transform
DWT	Discrete Wavelet Transform
FCM	Fuzzy C-Means
FFTs	Fast Fourier transformations
FN	False Negative
FNR	False Negative rate
FOA	Forest Optimization Algorithm
FP	False Positive
FPR	False Positive rate
GSC	Global seeding change
GMM	Gaussian Mixed Models
HH	High High
HL	High Low
HPF	High Pass Filter
HSI	Hue, Saturation, Intensity
HSV	Hue, Saturation, Value
Int-to-Int-HLWT	Integer Haar Lifting Wavelet Transform
LH	Low High
LL	Low Low

LSC	Local Seeding Changes
LWT	Lifting Wavelet Transform
MCC	Matthews Correlation Coefficient
PSO	Practical Swarm Optimization
RGB	Red, Green, Blue
SVM	Support Vector Machines
TN	True Negative
TNR	True Negative rate
TP	True Positive
TPR	True Positive rate
YCbCr	Luminance, Chrominance-blue, Chrominance-red
YUV	Luminance, two Chrominance components

List of Symbols

c	Cluster
Cb-mean	Chrominance-blue mean
Cr-mean	Chrominance-red mean
C_x and C_y	Principal point of camera
e	Gray level
f_x and f_y	Focal length of camera
GG	Number of gray levels
H_m	Histogram's partial entropies
i	Number of frames in video
J, K	Frame size
Lb	Lower band
m	Number of thresholds
N	New color space
N_p	Number of particles
N_S	Number of the selected frame
P_e	Probability histogram
Reg_{color}	Region of color
Reg_{fire}	Region of fire
Rot	Rotation matrix
t	Iteration Value
thr	Threshold
T-max	Maximum Iteration Value
Tr	Transformation matrix
Ub	Upper band
W	Inertia or momentum

W_m	histogram's partial probabilities
Y-mean	Luminance mean
Z	Video length
σ	Variance
μ	Mean
(Xrw, Yrw, Zrw)	Position fire in real world

Chapter One: Introduction

1.1 Overview

Climate change and temperature increases have significantly impacted the environment in recent years. These increasing temperatures directly effect to occur fires. The impact of fires on local and global ecosystems is severe, resulting in damage to infrastructure, injuries, and the loss of human, plant, and animal lives.

The Iraqi government statistics of fire incidents in the different regions of Iraq reached more than 32,477 fire incidents in 2022 (Iraqi Ministry of Interior 2023). These fires happen indoors, such as in restaurants, hotels, government offices, and houses, and outdoors such as in farms, forests, and wheat fields. Therefore, many economical and life losses as the result of these fires. Early fire detection and localization are very important issues and are requested for indoor and outdoor areas. Efforts have been exerted to curb the negative effects of fires by identifying them early on and creating maps of fire-prone areas (Pradhan, Suliman, and Awang 2007).

1.2 The Methods of Fire Detection

Traditional fire protection for outdoor areas uses mechanical equipment and tower monitoring by humans. While for indoor areas such as houses, buildings, and offices, the most common fire detection method is temperature sampling, particle sampling, and air transparency testing. The fire alarm is not activated unless the particle of smoke reaches the sensor roof. The most common fire detection are listed below.

1.2.1 Fire Watch Tower

In watch towers, it is crucial for people to constantly monitor the surroundings and promptly report any fires that occur. However, human observation can be affected by factors such as operator fatigue, time of day, season, and location, which may compromise the accuracy of detecting fires.

1.2.2 Wireless Sensor Networks

The conventional sensors that detect heat, flame, smoke, and gas usually take time to activate particles when reaching the sensor point. Furthermore, these sensors have relatively narrow ranges, and many must cover large areas. As well as, charging the battery is a great challenge. Also, it is required high cost.

1.2.3 Satellite and Aerial

Satellite-based monitoring systems can monitor large areas, but satellite images are low resolution. Since fire is only discovered after it has spread considerably, real-time detection is impossible. These systems are also highly pricey. The extended scanning time and limited resolution of satellites would severely reduce the accuracy of satellite-based forest fire monitoring due to weather conditions (such as clouds).

1.2.4 Video-based Smoke and Flame Detection

Video-based smoke and flame detection is an advanced technology that uses video cameras and intelligent algorithms to identify and locate smoke and flames in real-time. By analyzing the visual characteristics of the video feed, this technology offers a more reliable and comprehensive approach to fire detection, especially in large and open spaces. The combination of color and motion detection algorithms enables accurate and efficient identification of potential fire incidents. Video-based smoke and flame detection systems provide improved coverage and early detection, leading to prompt response and minimizing fire-related risks.

1.3 Literature Review

In recent years, several works have been proposed in the field of fire detection based on the vision system. As with the current relatively new subject of vision research, it is progressing and producing promising results.

This section provides an overview of the state-of-the-art technology for fire detection methods and fire localization algorithms. The review of fire detection is divided into three sub-sections based on a group of methods:

1.3.1 Color Detection of Fire

Color detection was the first technique used for video-based fire detection and is still used in most detection methods. Most color-based fire detection methods in video-based fire detection use *RGB* color space and sometimes combine (*Hue-Saturation-Intensity/Hue-Saturation-Value)* (J. Chen, He, and Wang 2010; T. C. Chen, Wu, and Chiou 2004; Günay et al. 2010; Yang et al. 2010). *RGB* format is commonly used in visible camera sensors as it displays the spectrum content corresponding to this color space, which is the primary reason for its usage in detecting video.

Phillips et al. (Phillips, Shah, and Da Vitoria Lobo 2002) used *RGB* color space and temporal variation. The Gaussian smooth color histogram detects the fire color pixel, then uses temporal changes in the pixel to determine which of these pixels is actually the fire color. The flame detection rate is 93.5%.

Liu et al. (Liu and Ahuja 2004) interduced Gaussian distributions in *HSV* color spaces to model fire colors and Fast Fourier transformations to describe fire contours for fire detection. The detection rate is 0.999.

Toreyin et al. (Toreyin, Dedeoglu, and Cetin 2006) utilized a hidden Markov model and wavelets to examine a recurring pattern in smoke boundaries. The behavior of the chrominance components *U* and *V* in the *YUV* color space was investigated, revealing a tendency for their decrease in a grayish scene with smoke. To model a smoke flicker, they studied the high-frequency behavior of the smoke boundaries. The delay to detect flame in frames is 97 frame with distance 25m.

Çelik et al. (Çelik, Özkaramanl, and Demirel 2007) studied different video sequences and images and presented fuzzy color models using statistical analysis *YCbCr/RGB*. Combined with spectral analysis and fire flicker, it is a property of detecting the existence of flame or smoke in the scene. The model enables a good distinction between fire and objects similar to fire. The detection rate is 99%.

Qi and Ebert (Qi and Ebert 2009) presented a cumulative fire matrix defined by combining *RGB* color and *HSV* saturation. Starting with the assumption that the green component of fire pixels changes widely compared to the red and blue components, this method evaluates the variation in spatial color in the pixel value to distinguish between non-fire-moving objects and uncontrolled fires. The system is able to correctly detect 60 kinds of fire videos. Table 1.1 shows a brief comparison between the above-explained methods.

Table 1.1 State-of-the-art of fire detection methods based on color detection.

References	Flame detection	Smoke detection	Color detection	Remarks	Results
(Phillips, Shah, and Da Vitoria Lobo 2002)	✓		RGB	Used temporal variation	The flame detection rate is 93.5%

(Liu and Ahuja 2004)	✓		HSV	Used FFT	The detection rate is 99.9%
(Toreyin, Dedeoglu, and Cetin 2006)		✓	YUV	Used a hidden Markov model and wavelets	The delay to detect flame in frames is 97 frame with distance 25m.
(Çelik, Özkaramanl, and Demirel 2007)	✓	✓	YCbCr RGB	Fuzzy color models	The detection rate is 99%.
(Qi and Ebert 2009)	✓		RGB/HSV	cumulative fire color matrix	The system is able to correctly detect 60 kinds of fire videos

1.3.2 Motion Detection of Fire

The detection of moving objects is also widely used in video-based fire detection since flame and smoke are moving objects. Additional analysis of the video-moving regions is necessary to differentiate between movement caused by fire and that caused by ordinary moving objects. The well-known moving object detection algorithm is the Background Subtraction (Günay et al. 2010) (J. Chen, He, and Wang 2010) method, optical flow analysis (Kolesov et al. 2010) and temporal differencing (Lee and Han 2007). These can be used as a part of video-based fire detection systems.

Töreyin *et al.*(Töreyin et al. 2006) used a color model to identify possible fire locations, then use wavelet analysis in the spatial and temporal dimensions to assess high-frequency activity in the area. Moving pixels and regions are determined in the video by using a hybrid background estimation method. The detection rate is 100%.

Ko et al. (Ko, Cheong, and Nam 2009) suggested a fire detection approach based on Support Vector Machines (*SVM*). Since flames are often brighter and contrast more than nearby areas, a luminance map excludes non-fire pixels. In addition, a two-class *SVM* classifier with a radial basis function kernel and a temporal fire model is employed to finalize the fire pixels. Furthermore, it used the color information and temporal variation of the pixel. To detect a moving region, a modified hybrid background estimation method is used. The detection rate is 86.5%.

Han and Lee (D. Han and Lee 2009) presented a method for analyzing color and movement information in tunnels to detect flames and smoke. Smoke detection is based on extracting motion areas using background images, image motion history, and invariant moments. In contrast, flame detection is based on color information, the intensity of images, erosion, and dilation to remove the noise. The total detection performance of flame is 96.3%, while that of smoke is 91.2%.

Chen et al. (J. Chen, He, and Wang 2010) interduced a method for detecting flames based on multi-feature fusion. Therefore, a flame-flicker detection algorithm using the temporal and spatial properties of the flames, such as the normal movement of flame by extracting foreground objects that move with improved Gaussian mixture modeling methods and color clues, is used to detect fire in color video sequences. The algorithm is able to detect the fire in a very short time-less than 2 s from the start of the fire.

Truong and Kim (Truong and Kim 2012) interduced a four-stage approach for fire flame detection, which includes: utilizing an adaptive mixture of Gaussian models to detect moving regions, implementing the Fuzzy C-Means (*FCM*) algorithm to identify candidate fire regions based on their color compared to other moving regions, extracting special parameters from the tempo-spatial characteristics of the fire regions, and employing a Supper Vector Machine (*SVM*) to differentiate between fire and non-fire. The average true positives is 94.78%.

Shidik *et al.* (Shidik et al. 2013) combined a rule-based multicolor extraction (including *RGB*, *HSV*, and *YCbCr*) with background extraction to generate a fire segmentation area and utilize morphological operation and time frame selection. Implementing this method can reduce the erroneous detection of the fire zone, resulting in an error rate decrease of approximately 15.2%. The drawback of this paper is that the algorithm has not been tested in real-time.

Foggia et al. (Foggia, Saggese, and Vento 2015) used *YUV* color space, morphological change, and motion assessment modules to distinguish each candidate region and designed a balanced vote strategy for a comprehensive decision. Although insufficient, supplementary color features, including textures, shapes, and optical flows, can reduce false detection. The accuracy of the system 93.55%.

Han *et al.* (X. F. Han et al. 2017) used multicolor (*RGB*, *HIS*, and *YUV*) models and Gaussian mixture models to detect motion. The results were good in the laboratory for detecting the flame, where the average detection rate was approximately 96%. On the other hand, Gaussian and color models cannot be applied to real-time scenarios, as they require considerable processing effort.

Gong *et al.* (Gong et al. 2019) presented a method for identifying fires based on several fire characteristics, which extracted the suspected fire area by identifying the motion via frame differences method and the images' color. In additional, they calculated the mass center of the fire in each frame to identify the fire by extracting the shape, spatial, and area variability of the images to improve identification accuracy. The experimental results show that the preposed methodology improves accuracy and reduces false positive rates but is not widely used.

Gagliardi and Saponara (Gagliardi and Saponara 2020) used multi-steps to smoke detection. The Kalman filter is used for motion detection, segmentation of the color (by transforming *RGB* frames to *HSV*), time-based blob analysis, blob labeling, and alert generation if at least 7 overlapping bounding boxes are observed. The method's performance is assessed using several datasets. The approach has been demonstrated to be highly effective across the majority of datasets, consistently achieving a 100% recall rate when compared to alternative methods.

Gagliardi et al. (Gagliardi, de Gioia, and Saponara 2021) conducted research using image processing and deep learning techniques to locate and identify smoke in images and movies. This technique uses a Kalman filter for motion detection, color segmentation, the extraction of bounding boxes around a gray object in motion, and prediction, which is carried out with the assistance of a Convolution Neural Network *CNN*. The hite rate of method is 90.49%.

Khalil *et al.*(Khalil et al. 2021) interduced multi-space color models and motion detection to detect fire objects with fewer parameters. The work explored the *RGB* and *LAB* color spaces for separating fire and regions similar to fire. In order to eliminate regions similar to fire, Gaussian Mixed Models *GMMs* are used to detect moving (fire) objects. The *GMM* model has a high performance compared to other models. However, the detection accuracy is high, the false positive rate remains high (88.81%).

Wahyono *et al.*(Wahyono et al. 2022) used fire color characteristics by developing probabilistic models using Gaussian multiples. In addition, other fire characteristics, i.e., dynamic fire movements modeled with motion characteristics based on moment-invariant, have also been applied. In the experiment, the true positive rate was 89.92 %, which is relatively high. However, one of the biggest challenges in implementing a module is that it may be difficult to install a camera physically. Table 1.2 shows a brief comparison between the above-explained methods.

Table 1.2 State-of-the-art of fire detection methods based on motion and color detection

References	Flame detection	Smoke detection	Color detection	Motion detection	Remarks	Results
(Töreyin et al. 2006)	✓		RGB	hybrid background estimation method	wavelet analysis in temporal and spatial	The detection rate is 100%
(Ko, Cheong, and Nam 2009)	✓		RGB	modified hybrid background estimation method	Luminance map	The detection rate is 86.5%
(D. Han and Lee 2009)	✓	✓	RGB	motion history, invariant moments	Experiment study for tunnel fire	The total detection performance of flame is 96.3%, while that of smoke is 91.2%.
(J. Chen, He, and Wang 2010)	✓		RGB/HSI	Gaussian mixture model0	Multi-feature fusion	The algorithm is able to detect the fire in a very short

						time-less than 2 s from the start of the fire.
(Truong and Kim 2012)	✓		FCM+CIE LAB	Gaussian mixture models	Used SVM	The average true positives is 94.78%.
(Shidik et al. 2013)	✓		RGB/HSV / YCbCr	Background subtraction	Time frame selection	The error rate is decrease approximately 15.2%.
(Foggia, Saggese, and Vento 2015)	✓		YUV	Optical flow	Real-time	The accuracy of the system 93.55%.
(X. F. Han et al. 2017)	✓		RGB/HSI/ YUV	Gaussian mixture models		The average detection rate is about 96%
(Gong et al. 2019)	✓		RGB/HSI	Frame differences	Real-time	accuracy and reduces false positive rates
(Gagliardi and Saponara 2020)		✓	HSV	Kalman estimator	geometrical features analysis	The recall of the method is 100% when compared with other methods
(Gagliardi, de Gioia, and Saponara 2021)		✓	HSV	Kalman filter	CNN classifier	The hite rate of method is 90.49%
(Khalil et al. 2021)	✓		RGB/LAB	Gaussian mixture models	Fire growth, Static	high performance

					object tracking	compared to other models
(Wahyono et al. 2022)	✓		RGB/HSV / YCbCr	moment invariants	Real-time	The true positive rate is 89.92 %

1.3.3 The Fire Localization

Most of the algorithms were focused on fire detection, but a few focused-on fire localization, so this section shows the prior methods of fire localization based camera.

Verstockt *et al.* (Verstockt et al. 2011) developed a method for calculating 3D models of fire and smoke using cameras distributed around the fire. This framework combines multiple cameras' single-view detection results with homogeneous projections on several horizontal and vertical planes that cut the scene. The virtual sensor points create a 3D grid when these slices are crossed. The location, size, and direction of the fire are accurately estimated. This approach only applies to fire fronts that are no more than 2×4 m^2. One of the most important aspects of fire extraction is detecting and extracting the fire area. Achieving this task in an unstructured outdoor setting poses a challenge.

Li *et al.* (Yimang Li et al. 2023) interducrd a real-time fire detection and localization for indoor places. A fully convolutional one-stage *CNN* is presented for fire detection. Two cameras specify the fire localization to locate the flame. They used two steps for the fire localization: the first step was for camera calibration from two frames. In contrast, the relative coordinates of the firing position to the anchor point are computed in the second step. Within 0.7 m, the localization accuracy could be achieved. The average error in localization was 0.44 meters. Table 1.3 shows a brief comparison between the above-explained methods.

Table 1.3 State-of-the-art of fire localization methods

References	Flame detection	Smoke detection	The method
(Verstockt et al. 2011)	✓	✓	The homography-based multi-view plane slicing is used for locating the 3D position and volume of the fire.
(Yimang Li et al. 2023)	✓		Two cameras for calibration and used anchor points to calculate the coordinate of the flame position to anchor points

1.4 Motivation

Despite its availability and lower cost, technology is not employed to control the huge number of fires in Iraq. Although some big buildings have sensors and cameras. However, open space monitoring has great challenges, especially in our environment, with the harsh sun, constant dust, and light change between night and day. It is necessary to propose an adaptive system utilizing the camera with continuous monitoring capable of addressing these challenges.

1.5 Problem Statements

The traditional methods of fire detection rely on sensors, but these sensors do not efficiently work in outdoor and huge indoor areas. This has led to the development of an alternative fire detection method based on closed circuit television (CCTV) cameras. These methods analyze the input video to detect the color and motion patterns that indicate a fire. This technology has the potential to detect fires in the early stages and provide a quicker time response for emergency services.

However, there are several challenges associated with implementing the fire detection-based camera, such as:

- The continuous variations in the luminance of the sun during the day.
- The effect of lighting on the flames detection system inside the building.
- Normally, these techniques require intensive algorithms that are unsuitable for real-time.
- A high false alarm can occur.
- The automatic localization of fire has few studies in the literature.

1.6 Thesis Objectives

The main objective of this work is to propose and design an efficient system for fire detection and localization that is suitable for real-time, and following specifications:

1. Adaptive detection for fire flames, colors can distinguish colors similar to the flame, such as the sunlight.
2. A multi-threshold detection for the smoke is produced from the fire, which can distinguish different densities of the smoke, even for very low density of smoke.
3. Localization of the fire position in open and huge areas.
4. High accuracy for fire detection, whether flame-based or smoke-based.

1.7 Contributions

The proposed work develops a computer vision system for fire detection and localization that can be easily employed in open areas, either indoors or outdoors. The scientific contributions to this research can be summarized as follows:

- Propose a fire detection system that has the ability to distinguish and solve the problem of the high color similarity between sunlight and fire flame. This distinguishing ability is designated based on adaptive light intensity thresholds.
- Also, the other system for fire smoke detection features has the ability to distinguish different smoke densities and colors; even very low-density smoke uses multi-thresholds to be an adaptive system for different smoke color detection.
- Find the approximate area of fire detection with localization of the fire position using simple algorithms that meet for real-time processing.

1.8 Thesis Outline

The remaining chapters of the thesis are organized as follows:

Chapter 2: Background and Theory

In this chapter, background and theory for image processing, clustering, and optimization are described in detail.

Chapter 3: The Proposed System of Fire Detection

The proposed system of fire detection with three models is presented along with localization for fire in the real world.

Chapter 4: Results and Discussions.

Includes a description of the results of the suggested methodology and comparisons with previous research.

Chapter 5: Conclusions and Future Works.

Includes the conclusions reached about fire detection and proposals for future work.

Chapter Two: Background and Theory

2.1 Introduction

Like other computer vision-related tasks, video-based smoke and flame detection are also composed of multiple steps that use computer vision technology at each stage. This chapter presents some techniques used for computer vision in image processing, clustering color, and optimization algorithms.

2.2 Digital Representation of Image

An image may be described as a two-dimensional function *f(x, y)*, where *x* and *y* are spatial coordinates, and *f* is the amplitude of a pair of coordinates, which at this time is called image intensity. When coordinates and amplitudes are digitalized, the image is called digital. The digitization of coordinate values is called sampling, and the digitization of amplitude is called quantization. Therefore, *x, y,* and *f* are discrete and finite quantities. A real numbers matrix will be the outcome of sampling and quantization. If an image *f(x, y)* were sampled to produce an image with *M* rows and *N* columns, the image size would be *M* × *N*, and the digital image might be represented as a matrix as in Equation (2.1). A digital image is represented by an array on the right side of this equation, and each item of that array is referred to as an image element (also known as a picture element or pixel) (Gonzalez, Woods, and Eddins 2009)

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \dots & f(0,N-1) \\ f(1,0) & f(1,1) & \dots & f(1,N-1) \\ \vdots & \vdots & & \vdots \\ f(M-1,0) & f(M-1,1) & \dots & f(M-1,N-1) \end{bmatrix} \quad \textbf{(2.1)}$$

2.3 Digital Image Types

The image processing toolbox can give different kinds of images:

- **RGB Images**

RGB colors are the *M* × *N* × *3* color pixel array, consisting of red, green, and blue at a certain spatial point.

- **Binary Images**

A binary image is a black-and-white representation of a logical array of 0s and 1s.

- **Gray Scale Images**

Gray scale images, also called grayscale or black and white images, are digital images that use shades of gray to represent the intensity or brightness of each pixel. The pixel values range from 0 (black) to 255 (white), with varying shades of gray in between.

- **Index Image**

Indexed images are composed of color maps and an image matrix, with color components specified in each row. Pixel values are computed by mapping them to the colors of the color map.

2.4 Types of Image Color Space Conversion

Almost all visible range cameras have sensors that can detect video in *RGB* format. Clear spectral content is associated with this color space, which is the major rationale for utilizing *RGB* (A Enis Çetin et al. 2013).

Despite the benefit of *RGB* color space, different color spaces exist because they represent color information in a way that makes certain calculations more convenient or provides a more intuitive way to recognize colors. For example, the *RGB* color space defines color as a percentage of red, green, and blue hues mixed. Other color models describe color in terms of hue (color shade), saturation (amount of gray or pure color), and lightness (intensity or overall brightness):

2.4.1 RGB Color Space to Gray Scale Conversion

To convert *RGB* values to grayscale values by taking a weighted sum of the *R*, *G*, and *B* components in Equation (2.2) (Wan and Xie 2016):

$$Grayscale = 0.299 * R + 0.587 * G + 0.114 * B \quad \textbf{(2.2)}$$

2.4.2 RGB Color Space to HSV Color Space Conversion

An image is referred to as an *m-by-n-by-3* numeric array with values in the range [0, 1]. Each pixel's hue, saturation, and value are determined by the third dimension of *HSV*. Hue is a value from 0 to 1 corresponding to a color's position on the color wheel. As the hue increases from 0 to 1, the color moves from red to orange, yellow, green, cyan, blue, magenta, and eventually back to red. Saturation refers to the amount of hue or deviation from neutral. A neutral shade is indicated by 0, while 1 represents maximum saturation. The value refers to the maximum value between a given color's red, green, and blue components (Image Types in the Toolbox - MATLAB & Simulink 2022). The *RGB* color model is inferior to the *HSV* color space. In portraying human perception and outlining the sense of color. In the *HEXCONE Model* (Smith 1978), the *RGB* values can be converted to the *HSV* color space using Equations (2.3) to (2.5). Figure 2.1 illustrates the *RGB* color space and *HEXCONE Model*.

$$V = \max\,(R, G, B) \quad \textbf{(2.3)}$$

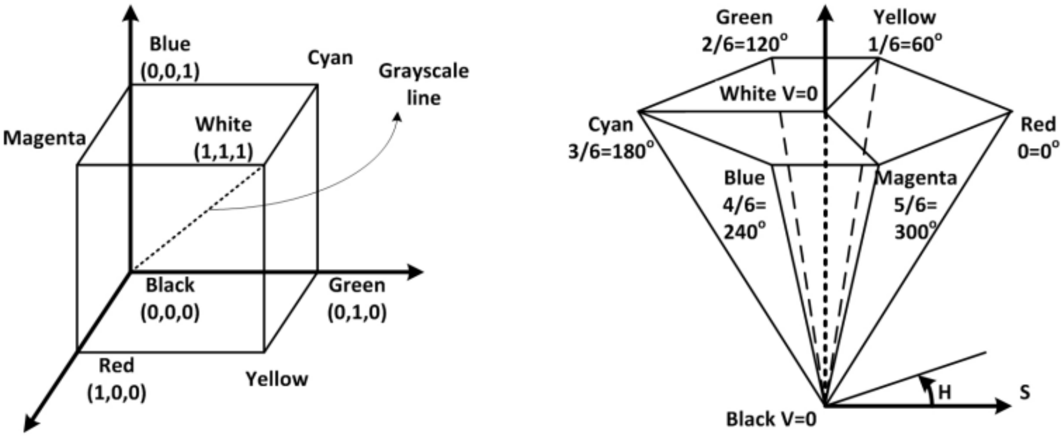
$$let\colon X = \min\,(R, G, B)$$

$$S = (X - V)/V \quad \textbf{(2.4)}$$

$$let\!: r = (V - R)/(V - X)$$
$$g = (V - G)/(V - X)$$
$$b = (V - B)/(V - X)$$

$$If\ R = V\ then\ H = (if\ G = X\ then\ 5 + b\ else\ 1 - g)$$
$$If\ G = V\ then\ H = (if\ B = X\ then\ 1 + r\ else\ 3 - b)$$
$$else\ H = (if\ R = X\ then\ 3 + g\ else\ 5 - r) \qquad \textbf{(2.5)}$$

$$H = H/6$$



(a) (b)

Figure 2.1 RGB cube and HSV hexcone representations (a.) RGB. (b.) HSV

2.4.3 RGB Color Space to YCbCr Color Space Conversion

The *YCbCr* color scheme is one of the most popular color schemes used to define digital video components. A color scheme represents the color as a luminance and two-color-difference signal, where *Cb* and *Cr* represent the Chrominance-blue, and Chrominance-red components, respectively. The advantage of the *YCbCr* color scheme is a better distinction between lightness and Chrominance.

RGB color spaces can be used to identify different colors, but the drawback is that it is light-dependent. As a result, color rules may not function accurately when lighting conditions change. To resolve this, conversion of the

RGB color space to a color space with a clearer distinction between chrominance and intensity is necessary. The following formula is used to convert *RGB* color space to *YCbCr* color space (Celik et al. 2007) in Equation (2.6):

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0.2568 & 0.5041 & 0.0979 \\ -0.1482 & -0.2910 & 0.4392 \\ 0.4392 & -0.3678 & -0.0714 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 16 \\ 128 \\ 128 \end{bmatrix} \qquad \textbf{(2.6)}$$

where *Y* is luminance, *Cb* is Chrominance-blue, and *Cr* is Chrominance-red components.

2.4.4 RGB Color Space to CIE L*a*b Color Space Conversion

The International Commission on Illumination, also known as *CIE*, invented the device-independent color spaces (*CIE 1976 XYZ* and *CIE 1976*) *L*a*b**. These color spaces simulate hues using the three kinds of cone cells in the human eye, which are often sensitive to different hues.

The basic model produced by the *CIE* is the *XYZ* color space. The *Y* channel represents the color's luminance brightness. The *Z* channel roughly refers to how much blue is in the image, but *Z's* value in the *XYZ* color space is not the same as *B's* value in the *RGB* color space. This is because the *X* channel in the *XYZ* color space does not have a corresponding color attribute; it is represented by an axis orthogonal to the *Y* brightness and *Z* axes in the 3D coordinate system.

The *L*a*b** color space offers a uniform representation of colors compared to the *XYZ* model. Some colors in the *L*a*b** color space may not be representable in the *RGB* color space (the set of valid *RGB* colors). For example, if the *L*a*b** values [100, 100, 100] are converted to *RGB* color space, the returned values are [1.7682, 0.5746, 0.1940] is not an acceptable *RGB* color.

*L** refers to the image's brightness or luminance. Values fall between [0, 100], where (0) designates black and (100) designates white. Colors get brighter as *L** rises; *a** stands for the percentage of red or green tones in the image. Red or magenta is represented by a high positive *a** value. Green has a significant negative *a** value. Although *a** does not have a single range, values frequently lie between [-100, 100] and [-128, 127), and *b** represents the proportion of yellow or blue tones in the image. Yellow is represented by a high positive *b** value. Blue is represented by a significant negative *b** value. Although *b** does not have a single range, values frequently lie between [-100, 100] and [-128, 127) (Understanding Color Spaces and Color Space Conversion - MATLAB & Simulink - MathWorks Switzerland 2022). The transformation method from the *RGB* color space to *L*a*b** is a direct model method (León et al. 2006), which includes two steps with Equations (2.7) and (2.8).

The first step is the transformation of *RGB* → *XYZ* color space:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} M_{11} & M_{12} & M_{13} & M_{14} \\ M_{21} & M_{22} & M_{23} & M_{24} \\ M_{31} & M_{32} & M_{33} & M_{34} \end{bmatrix} \begin{bmatrix} R \\ G \\ B \\ 1 \end{bmatrix} \qquad \textbf{(2.7)}$$

The second step is the transformation of *XYZ* → *L*a*b** color space:

$$L^* = \begin{cases} 116 \times \left(\frac{Y}{Y_n}\right)^{\frac{1}{3}} - 16\,, & if \ \left(\frac{Y}{Y_n}\right) > 0.008856 \\ 903.3 \times \left(\frac{Y}{Y_n}\right), & if \ \left(\frac{Y}{Y_n}\right) \leq 0.008856 \end{cases}$$

$$a^* = 500 \times \left\{\left(\frac{X}{X_n}\right)^{\frac{1}{3}} - \left(\frac{Y}{Y_n}\right)^{\frac{1}{3}}\right\} \qquad \textbf{(2.8)}$$

$$b^* = 200 \times \left\{\left(\frac{Y}{Y_n}\right)^{\frac{1}{3}} - \left(\frac{Z}{Z_n}\right)^{\frac{1}{3}}\right\}$$

Mij is a linear transformation matrix between the spaces *RGB* and *XYZ*, and *Xn*, *Yn*, and *Zn* are the values of the reference blank.

2.5 Image Preprocessing

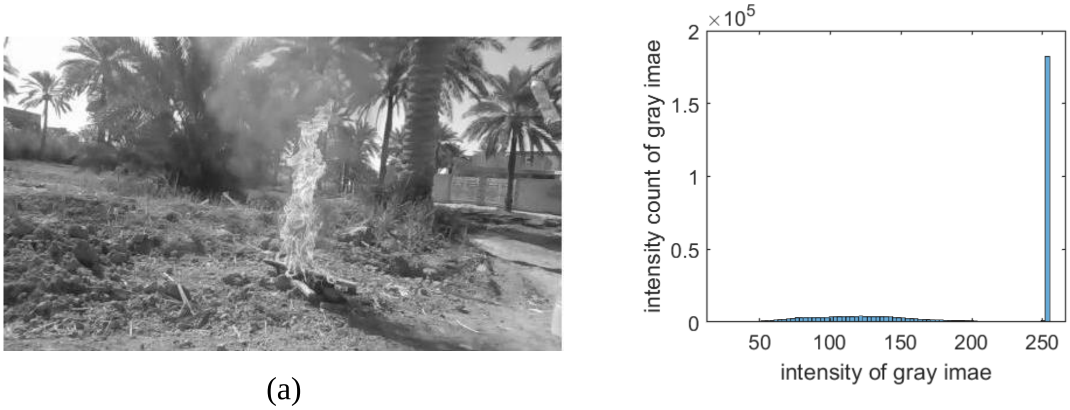
Directly collected sensor images often contain incomplete and inconsistent noise, which must be preprocessed to achieve correct classification. The basic preprocessing methods are sampling, discretization, noise elimination, transformation, and integration (KC and Nattee 1970). Preprocessing images is similar to the scientific standardization of data sets; this is a general step in many descriptive techniques of features. The preprocessing of the image is used to process the degradation of the image. It is, therefore, necessary to obtain some previous data or detailed information, such as details of the degree of degradation, characteristics of the imaging system, and the environment in which the images were taken (Chaki and Dey 2019). Some complications can be resolved by using image preprocessing techniques:

2.5.1 Image Intensity Histograms

In image processing, intensity transformation functions, which are based on data obtained from image intensity histograms, play a crucial role. Equation (2.9) defines the histogram of a digital image with L total potential intensity levels in the range $[0, G]$, where r_k is the k_{th} intensity level in the interval $[0, G]$, and n_k refers to the number of pixels in the image with intensity level r_k (Gonzalez, Woods, and Eddins 2009)

$$h(r_k) = n_k \tag{2.9}$$

An image histogram is a graph that show the distribution of intensities in a grayscale image or index. In the histogram, details can be used to select the appropriate amelioration operation. An example would be if an image histogram displays a limited range of values, use a function to spread the values over a larger range. By default, Figure 2.2 displays an image with a histogram. The histogram displays a peak at 251, which corresponds to the dark gray background in the image (Image Processing Toolbox Documentation 2022). There are various ways to represent histograms, such as bar charts, plot, and stem (Gonzalez, Woods, and Eddins 2009).



(a)

(b)

Figure 2.2 Histogram of images (a.) grayscale image (b.) image histogram (Image Processing Toolbox Documentation 2022)**.**

2.5.2 Image Filtering

Filtering is an image modification or enhancement technology. The image filter can be used to highlight or delete some features.

The median filter is nonlinear. The values of a relative pixel's nearby pixels are used to conditionally determine the difference in the nonlinear filtering. It works by sorting pixel values in a range, averaging and changing the median of the range between the original values (Marques 2011) and (Chaki and Dey 2019). The median filter is very good at reducing "salt and pepper" noise (a form of noise that produces very bright so-called salt and very dark so-called pepper outliers in an image). Figure 2.3 shows the median filter of images with pepper and salt noise.

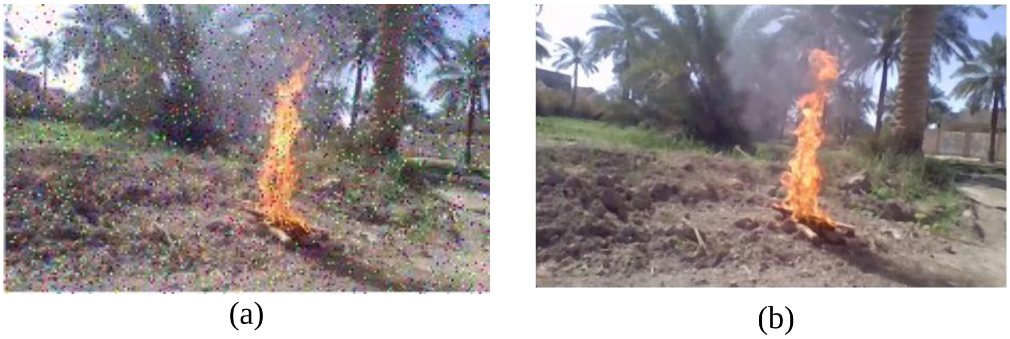

(a) (b)

Figure 2.3 Image filtering (a.) image with salt and pepper noise (b.) The noise elimination by median filter.

2.6 Wavelet Transform

The wavelet transform is a method that uses a wavelet function (called the "mother wavelet") to decompose an image into lower resolution levels (Torrence and Compo 1998). This is done by adjusting the scaling and shifting factors of the wavelet function. A wavelet function has a wave shape, a limited but flexible length, and a zero mean value that is localized in both the time and frequency domains. The wavelet transformation of an image generates a wavelet coefficient set at different scales (Nalley, Adamowski, and Khalil 2012). The high scale refers to the extended version of the wavelet, and the corresponding wavelet coefficient provides information about the low-frequency components of the image. The lower scales of a mother wavelet function represent a compressed version of the wavelet and are used to identify the high-frequency components of the analyzed signal. Since the wavelet transform can be used to analyze time series data by breaking it down into shorter, medium, and longer periods and identifying the main components that influence the trend in the series (Kim 2004), this can help to understand and detect patterns in the data. There are two main approaches to the wavelet transform: the discrete wavelet transforms (*DWT)* and the continuous wavelet transform (*CWT)*.

The *CWT* works with smooth, continuous functions and can decompose and analyze images at all scales (Nalley, Adamowski, and Khalil 2012). On the other hand, the *DWT* is a method of analyzing discrete digital images using scales represented by integer powers of two (a dyadic arrangement). This makes the transformation process simpler and more efficient than the *CWT*, which generates a large amount of redundant information and is more difficult to implement (Percival 2008).

2.6.1 The Discrete Wavelet Transform (DWT)

The *DWT* is a method for decomposing an input image *x(n)* into low and high-frequency components. This is done by filtering the image with a low-pass filter, *L*, and a high-pass filter, *H*. The outputs of these filters are then sub-sampled by two to produce the low-frequency (approximation, *a*) and high-frequency (detail, *d*) coefficients. These filters are known as the decomposition filter bank, and the method is called the convolution-based *DWT* approach (Mallat 1989).

To reconstruct the image, the low- and high-frequency coefficients (*a* and *d*) are up-sampled by inserting zeros between each pair of samples. Next, the up-sampled coefficients are filtered with reconstruction low-pass and high-pass filters (*L'* and *H'*), respectively. Finally, the outputs of these filters are added together to produce the reconstructed signal *x'(n)*. The reconstruction filter bank, consisting of filters (*L'* and *H'*), has inverse transfer functions compared to the decomposition filters *L* and *H*.

Similarly, for a multiresolution *DWT* decomposition, the low-pass sub-band *a* is further analyzed to find the second level of decomposition, and the process is repeated. The inverse of this process follows similar multi-stage synthesis filtering, reconstructing the output image by reconstructing low-pass *L* and high-pass filters *H* (Shahadi, Jidin, and Way 2013).

2.6.2 Integer Lifting Scheme Based on DWT

The lifting scheme, proposed by (Sweldens 1996), is a method for implementing the *DWT* that allows all operations to be executed in parallel, making it faster than the convolution-based *DWT*. The discrete wavelet transform *DWT* is calculated using the following three steps in the lifting scheme: splitting, predicting, and updating. The split step separates the input image *x(n)* into even and odd samples. The predict step estimates the value of the odd samples using the even samples to find the details coefficients, and the update step adjusts the even samples using the predicted values (Shahadi, Jidin, and Way 2013).

2.6.3 Integer Haar Lifting Wavelet Transform

The Haar filter is a bank filter commonly used with the *DWT*. In the Haar *DWT*, the approximation coefficients are obtained by taking the average of pairs of adjacent samples in the input image. The details coefficients are then calculated by finding the difference between these adjacent samples. Generally, the approximation coefficients (which represent the smooth components of the image) are similar to the original input samples due to the high correlations between neighboring samples in the input image. However, on the other hand, the details coefficients have lower power compared to the original image due to the same correlations (Shahadi, Jidin, and Way 2013).

The analysis of images of size ($n \times m$) using wavelet transform involves the classification of data (image) into two categories: estimation and relevant data (sub-image).

This process utilizes two types of filters, namely high-pass and low-pass filters, which are illustrated in Figure 2.4. Through these filters, the image is modified. Next, the image's foreground and background components can be segregated by considering their relative frequencies (approximation). This results in the generation of four distinct signals from each tier (Bamerni and Al-Sulaifanie 2019).

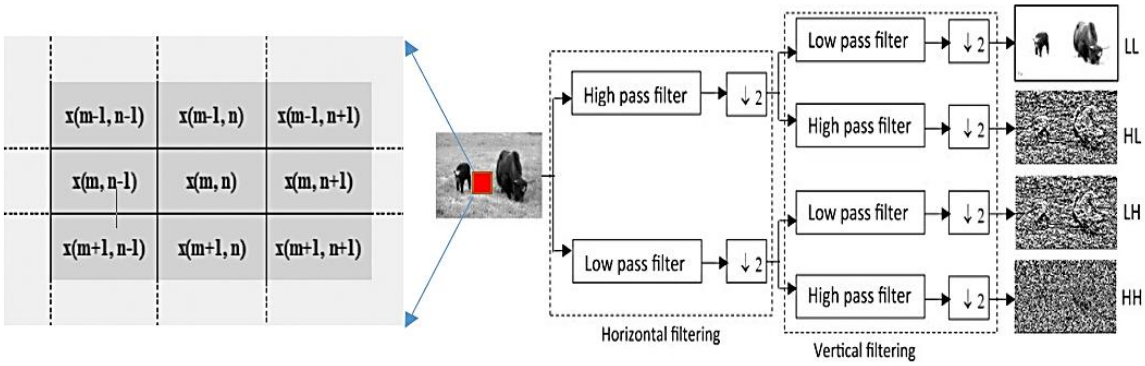


Figure 2.4 (2D) discrete wavelet transform decomposition image (DWT) (Parida and Bhoi 2017)

The Haar wavelet transform breaks down a image into four sets of coefficients, as illustrated in Figure 2.5 (a): the approximation coefficients (*LL*), horizontal detail coefficients (*HL*), vertical detail coefficients (*LH*), and diagonal detail coefficients (*HH*). The computation of these coefficients involves the calculation of the low-pass (*L)* and high-pass (*H)* subbands obtained through the application of the 1D Haar wavelet transform (Al Jumah 2013). The equations for calculating these subbands are as follows (Bamerni and Al-Sulaifanie 2019):

$$H_{11} = [X_{11} - X_{21}] \times \frac{1}{\sqrt{2}} = \frac{X_{11} - X_{21}}{\sqrt{2}} \qquad \textbf{(2.10)}$$

$$L_{11} = \left[X_{21} - \frac{1}{2} H_{11}\right] \times \sqrt{2} = \frac{X_{11} - X_{21}}{\sqrt{2}} \qquad \textbf{(2.11)}$$

In the same way

$$H_{12} = [X_{21} - X_{22}] \times \frac{1}{\sqrt{2}} = \frac{X_{21} - X_{22}}{\sqrt{2}} \quad \textbf{(2.12)}$$

$$L_{12} = \left[X_{21} - \frac{1}{2} H_{12}\right] \times \sqrt{2} = \frac{X_{21} - X_{22}}{\sqrt{2}} \quad \textbf{(2.13)}$$

To create the subbands *LL*, *LH*, *HL*, and *HH*, the 1D transform is applied in the row direction on both *L* and *H*.

$$LH_{11} = [L_{11} - L_{21}] \times \frac{1}{\sqrt{2}} = \frac{X_{11} - X_{21} + X_{12} - X_{22}}{2} \quad \textbf{(2.14)}$$

$$LL_{11} = \left[L_{12} + \frac{1}{2} LH_{11}\right] \times \sqrt{2} = \frac{X_{11} + X_{21} + X_{12} + X_{22}}{2} \quad \textbf{(2.15)}$$

$$HH_{11} = [H_{11} - H_{12}] \times \frac{1}{\sqrt{2}} = \frac{X_{11} - X_{21} - X_{12} + X_{22}}{2} \quad \textbf{(2.16)}$$

$$HL_{11} = \left[H_{12} + \frac{1}{2} HH_{11}\right] \times \sqrt{2} = \frac{X_{11} + X_{21} - X_{12} - X_{22}}{2} \quad \textbf{(2.17)}$$

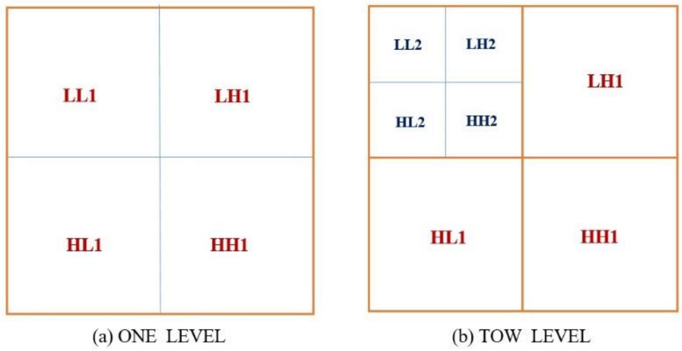


Figure 2.5 DWT image (a.) One-level (b.) Tow-level

In this scenario, all subbands necessitate the same set of sample components (*X₁₁*, *X₂₁*, *X₁₂*, *and X₂₂*) simultaneously, forming a 2 × 2 sample block. These sample components are sequentially moved line by line to cover the entire input matrix.

The Haar wavelet transform can be computed iteratively, commencing with the original signal as the approximation coefficients LL_o. At each decomposition level, the *LL* coefficients are further decomposed into the horizontal, vertical, and diagonal detail coefficients *HL*, *LH*, and *HH*, respectively, utilizing the aforementioned equations. This process is repeated until the desired level of decomposition is achieved (Anutam and Rajni 2006).

Figure 2.6 demonstrates the further decomposition of the LL_1 subband used to acquire the subsequent coarse level of wavelet coefficients by analytical sampling (Chandrasekaran 2021).

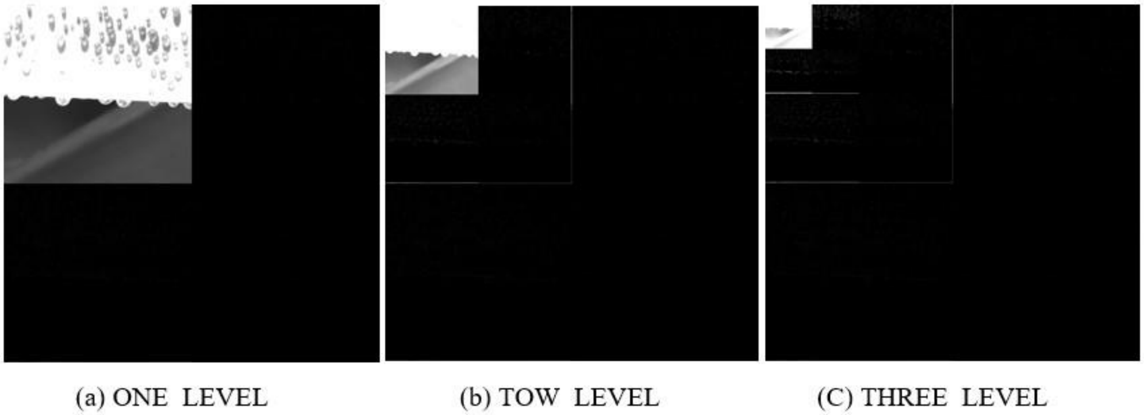


Figure 2.6 DWT with different–level (a.) one level (b.) two level (c.) three level

2.7 Thresholding

Thresholding is defined as the process of converting a grayscale image into a binarized image or an image with a new range of grayscales using a specified threshold (Araki et al. 2015) and (Chaki and Parekh 2011). Thresholding aims to extract some pixels from an image while removing others. The idea is to group similarly bright pixels from the first pixel with similarly bright pixels from the background pixel (Chaki and Dey 2019). The basic problem with a threshold is converting the image from several gray levels to a lower gray level, usually two levels. This conversion is usually performed when the pixel's intensity is compared to the reference value and replaced by a value, which means white or black according to the comparison results. Equation (2.18) is a mathematical representation of the threshold process, *f(x,y)* is an input image, *g(x,y)* is a binarized image, and *thr* is a threshold value (Marques 2011).

$$g(x,y) = \begin{cases} 1 & if\ \ f(x,y) > thr \\ 0 & \text{otherwise} \end{cases} \tag{2.18}$$

One method for determining a threshold is to examine the image histogram visually. Another way to select *thr* is by trial and error, choosing different thresholds until find that the observer's judgment gives a good result.

2.7.1 Otsu Algorithm

Otsu's method is a type of image threshold segmentation (OTSU 1979). It is an effective criterion for image processing to convert gray image to binary image.

The Otsu technique divides the image into two bright and dark areas, T_0 and T_1, where region T_0 is a set of intensity levels ranging from 0 to the threshold value, *l* is the maximum image region $T_1 = \{t, thr+1 \ldots, l-1, l\}$ and $\mathrm{T}_0 = \{0,1 \ldots .thr\}$ where *thr* represents the threshold value and *l* the image maximum gray level (for instance 256). T_0 and T_1 can be applied to both the object and the background or vice versa (the object does not always occupy the light region). The thresholding method developed by Otsu scans all potential thresholding values and determines the minimum value for each pixel level on either side of the threshold. The aim is to identify the threshold value for the sum of foreground and background with the lowest entropy. The variance of clusters T_0 and T_1 may be determined using Otsu's approach, which determines the threshold value based on the statistical information of the image. The best threshold value is determined by minimizing the sum of the weighted group variances, where the weights represent the individual groups' probabilities.

Given: $p(i)$ is the probability histogram of the observed gray value

$$i = 1, \ldots, l.$$

$$P(i) = \frac{number\{(r,c)|_{image(r,c)=i}\}}{(R,C)} \quad \textbf{(2.19)}$$

where *r*, *c* is the image's row, and column indexes and *R*, *C* is the image's number of rows and columns, respectively.

For intensity values ranging from *0* to *thr*, $w_b(thr), \mu_b(thr), and\ \sigma_b^2(thr)$ represent the weight, mean, and variance of class *T₀*. For intensity values ranging from $thr + 1$ to *l*, $w_f(thr), \mu_f(thr), and\ \sigma_f^2(thr)$ represent the weight, mean, and variance of class *T₁*. σ_w^2 as the summed weighted variances of the groups.

The lowest within-class variation value is the optimum threshold value, *thr** (Yousefi 2011). The variance inside a class is defined as follows:

$$\sigma_w^2 = w_f(thr) * \sigma_f^2(thr) + w_b(thr) * \ \sigma_b^2(thr) \quad \textbf{(2.20)}$$

where

$$w_b(thr) = \sum_{i=1}^{thr} P(i) \quad \textbf{(2.21)}$$

$$w_f(thr) = \sum_{i=thr+1}^{l} P(i) \quad \textbf{(2.22)}$$

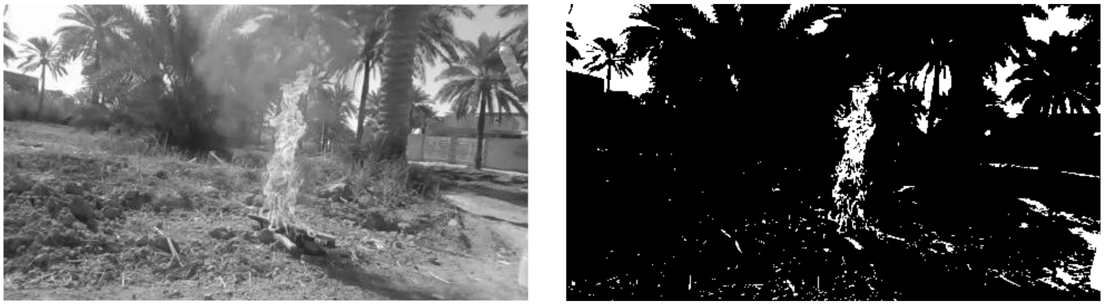
$$\mu_b(thr) = \frac{\sum_{i=1}^{thr} i * P(i)}{w_b(thr)} \quad \textbf{(2.23)}$$

$$\mu_f(thr) = \frac{\sum_{i=thr+1}^{l} i * P(i)}{w_f(thr)} \quad \textbf{(2.24)}$$

$$\sigma_b^2(thr) = \frac{\sum_{i=1}^{thr} (i - \mu_b(thr))^2 * P(i)}{w_b(t)} \quad \textbf{(2.25)}$$

$$\sigma_f^2(thr) = \frac{\sum_{i=thr+1}^{l}(i-\mu_f(thr))^2 * P(i)}{w_f(thr)} \qquad \textbf{(2.26)}$$

An illustration of Otsu thresholding is shown in Figure 2.7.



(a) (b)

Figure 2.7 Otsu’s thresholding (a.) original image (b.) result of Otsu’s method.

2.7.2 Kapur Algorithm

Another type of image threshold segmentation is Kapur's entropy (Kapur, Sahoo, and Wong 1985). Entropy-based thresholding is an effective segmentation approach based on the gray-level histogram's probability distribution. Entropy is at its highest when the optimal thresholds for distinguishing the classes are correctly allocated. Finding the best thresholds that produce the most entropy is the goal. It is common practice to calculate the entropy of a discrete source using the probability distribution *p = pi*, where pi is the probability that the system might exist in state *i* (Portes de Albuquerque et al. 2004). When normalized by the total number of gray levels *L*, the probability of each gray level *i* is equal to the gray level's relative occurrence frequency Equation (2.27).

$$p_i = \frac{h(i)}{\sum_{i=0}^{L-1} h(i)} \;, \quad i = 0, \ldots\ldots L\text{-}1 \qquad \textbf{(2.27)}$$

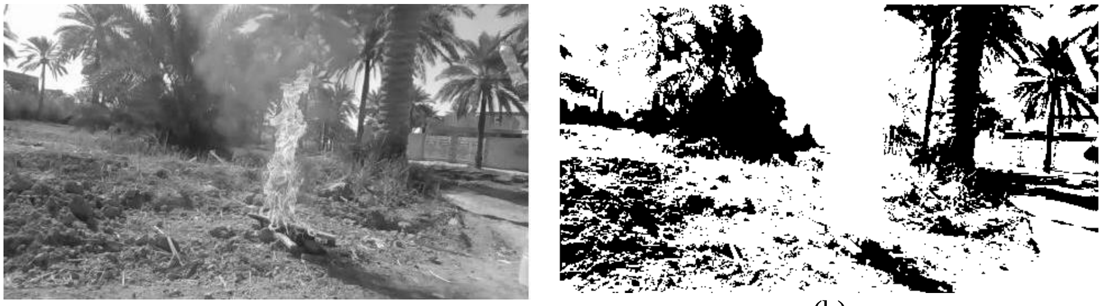
The separability and compactness of classes are gauged using Kapur's entropy. According to Equation (2.28), Kapur's entropy may be explained by bi-level thresholding.

$$H_0 = -\sum_{i=0}^{thr-1} \frac{p_i}{w_0} \ln \frac{p_i}{w_0}, \quad w_0 = \sum_{i=0}^{thr-1} p_i$$
$$H_1 = -\sum_{i=thr}^{L-1} \frac{p_i}{w_1} \ln \frac{p_i}{w_1}, \quad w_1 = \sum_{i=thr}^{L-1} p_i \qquad \textbf{(2.28)}$$

The threshold is optimal when the sum of class entropy is the maximum Equation (2.29).

$$thr^* = \arg\max\,(H_0 + H_1) \qquad \textbf{(2.29)}$$

An illustration of Kapur thresholding is shown in Figure 2.8.



(a)

(b)

Figure 2.8 Kapur thresholding (a.) original image (b.) result of Kapur method.

2.7.3 Threshold-Based Optimization Techniques

The most crucial step in image thresholding algorithms is choosing the optimal threshold value, while determining multiple thresholds is a common problem. One threshold value divides the image into two classes in bi-level thresholding techniques: foreground and background, but the bi-level thresholding method is ineffective when the image is quite complex and contains a variety of objects (Díaz-Cortés et al. 2018; Ewees, Abd Elaziz, and Oliva 2018). As a result, multilevel thresholding methods are frequently utilized for image segmentation (Ashish Kumar Bhandari et al. 2014; Manikandan et al. 2014). In recent years, numerous thresholding methods have been proposed, including Otsu's method (OTSU 1979), Kapur's entropy

(Kapur, Sahoo, and Wong 1985), minimum cross-entropy (J. Li et al. 2019), fuzzy entropy (Pare et al. 2018), and Tsallis entropy (A. K. Bhandari, Kumar, and Singh 2015). These methods enable multilevel thresholding segmentation through the use of optimization algorithms. For example, for the maximum entropy criterion, Horng and Jiang (Horng and Jiang 2010) introduced a firefly optimization algorithm, while Maitra and Chatterjee (Maitra and Chatterjee 2008) used a particle swarm optimization algorithm to create a hybrid cooperative in-depth learning model. Additionally, Sathya and Kayalvizhi (Sathya and Kayalvizhi 2011b) developed a bacterial foraging algorithm to enhance Otsu's minimal variance criteria and the maximum entropy criterion. For the maximum entropy and minimum variance criteria, Sathya and Kayalvizhi (Sathya and Kayalvizhi 2011a) proposed a modified bacterial foraging algorithm. Oliva and others (Oliva et al. 2013) suggested multilevel image thresholding for the related approaches based on the harmony search optimization algorithm. The maximum entropy criterion was then optimized using the artificial bee colony (ABC) algorithm, as described by Horng (Horng 2011).

2.7.3.1 Particle Swarm Optimization (PSO) Algorithm

Kennedy presented the *PSO* in 1990 (Eberhart and Kennedy 1995) to emulate the natural swarming of fish and birds. A collection of particles moving through search space in *PSO* is called a swarm. In order to locate the global optimal, the particles shift their positions following their prior experience and the swarm's best experience. It makes use of the swarm particles' global communication and real-number randomness. The *PSO* begins by generating a random population, after which the velocity and current position of all particles are updated for each iteration as described in the following equations:

$$v_i(t+1) = wv_i(t) + R_1C_1\left(P_i^{best} - x_i\right) + R_2C_2\left(g_i^{best} - x_i\right) \quad \textbf{(2.30)}$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad \textbf{(2.31)}$$

At each iteration *t*, the position and velocity of each particle are represented by x_i and v_i, respectively. R_1 and R_2 are the random values of two integers between 0 and 1, *w* is a weight that determines the particle's inertia, and C_1 and C_2 are factors that control the influence of the personal best and global best positions on the particle. P_i^{best} represents the best position found by the *ith* particle. At last, g^{best} is the best position in the population discovered thus far. Algorithm 2.1 provides the *PSO's* steps (Ewees, Abd Elaziz, and Oliva 2018):

Algorithm 2.1

```
Input: the size of population $N$, the values of $w$, $R_1$, $R_2$, $C_1$, and $C_2$.
Generate a random population $x$ and its velocity $v$.
The current iteration $t = 1$.
Output: optimal solution for objective function
while criterion do
for all n particles do
Compute the new velocity $v_i(t+1)$ using Equation (2.30).
Update the position using Equation (2.31).
Evaluate objective function $f_i[x_i(t+1)]$ of $x_i(t+1)$.
Find the minimum or maximum function of the current particle.
end for
Find the global fitness function $f_g^{best}$ and its corresponding position $g^{best}$.
```

```
t = t+1.
end while
Output: $f_g^{best}$ and $g^{best}$.
end.
```

2.7.3.2 Forest Optimization Algorithm (FOA)

The Forest Optimization algorithm is a process of evolutionary algorithm inspired by several trees in the forest (Ghaemi and Feizi-Derakhshi 2014). To solve problems in continuous search space, *FOA* is proposed. The *FOA* process consists of three main phases: "local seeding of the trees, population limiting, and global seeding of the trees." Figure 2.9 depicts the *FOA* flowchart. FOA's starting point is the initial population of trees (solutions) that make up this algorithm's forest. Each tree depicts a possible solution to the issue. In addition to its values, a tree has an attribute representing its age. This attribute, called "Age," is used to track the age of the association tree. Each newly generated tree has its "Age" set to "0" (Ghaemi and Feizi-Derakhshi 2014).

When trees begin seeding, some seeds fall just beneath the parent tree, where they mature into young trees (Ghaemi and Feizi-Derakhshi 2014), which is simulated in *FOA* through local seeding. The local seeding stage will operate on trees with "Age" ‘0’ following the initialization of the trees in order to simulate the nearby seeds of the parent trees. The trees then age, and their "Age" increases by one, except for newly generated trees. This stage is a simulation of the algorithm's local search.

The next step is population limiting, in which trees whose "Age" value is greater than their "lifetime" value are removed from the forest and become the candidate population (Ghaemi and Feizi-Derakhshi 2014). The remaining forest trees are also sorted according to their fitness value during the population-limiting stage. If the total number of forest trees exceeds the pre-defined "area limit" parameter, additional trees will also join the candidate population.

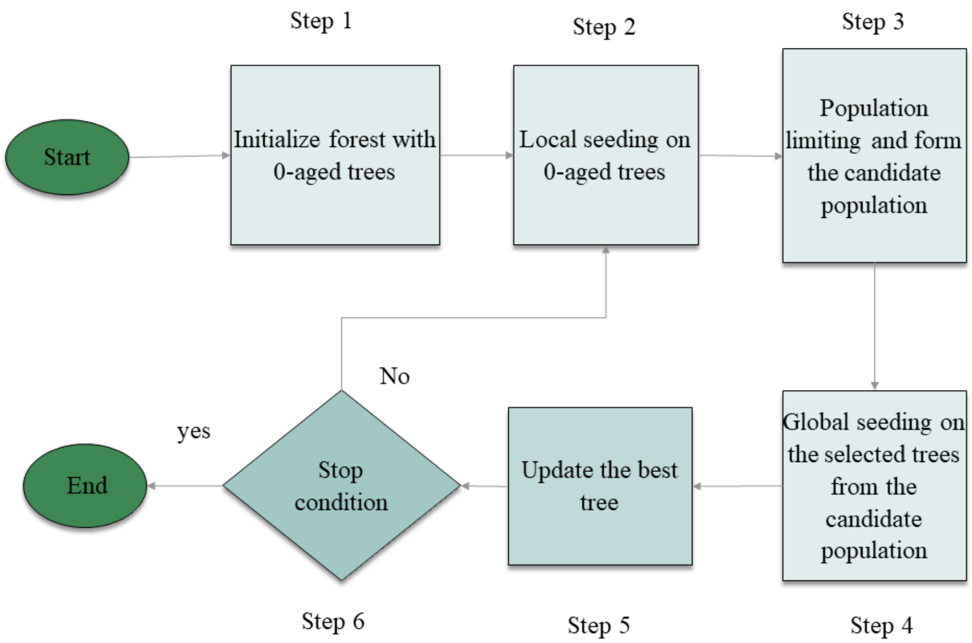

Step 1
Step 2
Step 3
Start
Initialize forest with 0-aged trees
Local seeding on 0-aged trees
Population limiting and form the candidate population
No
yes
End
Stop condition
Update the best tree
Global seeding on the selected trees from the candidate population
Step 6
Step 5
Step 4

Figure 2.9 Flowchart of FOA (Ghaemi and Feizi-Derakhshi 2014)**.**

Finally, the percentage of the candidate population is chosen through the global seeding phase. The selected trees from the candidate population will be utilized during the global seeding stage. The global search for *FOA* is modeled after the global seeding stage (Ghaemi and Feizi-Derakhshi 2014). The next step in *FOA* is to update the best tree by selecting the best solution based on its fitness value, setting its "Age" to 0 to prevent aging, and removing the best tree from the forest. These phases will progress iteratively until the termination requirement is satisfied. The Forest Optimization Algorithm has five parameters that must be set up at the beginning (Ghaemi and Feizi-Derakhshi 2014):

1. “Local Seeding Changes,” also known as "*LSC*."
2. The forest's limitation or "area limit."
3. The maximum allowed “Age” of a tree is named the “lifetime” parameter.
4. The "transfer rate," or percentage of the candidate population, will be used in the global seeding stage.
5. "Global Seeding Changes" or "*GSC*" is the number of variables whose values will change during the global seeding stage.

Algorithm 2.2 provides the *FOA’s* steps (Ghaemi and Feizi-Derakhshi 2014):

Algorithm 2.2

```
Algorithm FOA (lifetime, LSC, GSC, transfer rate, area limit)
 Input: lifetime, LSC, GSC, transfer rate, area limit
 Output: near-optimal solution for objective function f(x)
 Initialize the forest with random trees
    Each tree is a (D+1)-dimensional vector x, x= (age, $x_1$, $x_2$.... $x_D$) a D-dimensional problem
    The "age" of each tree is initially zero
  While the stop condition is not satisfied, do
    Perform local seeding on trees with age 0
            For =1: "LSC"
                Randomly choose a variable of the selected tree
                Add a small amount dx- dx ε [-Ax, Ax]- to the randomly selected variable
                Increase the age of all trees by 1 except for new generated trees in this stage
    Population limiting
```

```
            Remove the trees with ages bigger than "life time" parameter and add them to the candidate population
            Sort trees according to their fitness value
            Remove the extra trees that exceed the "area limit" parameter from the end of the forest and add them to the candidate population
    Global seeding
            Choose "transfer rate" percent of the candidate population
            For each selected tree
                Choose "GSC" variables of the selected tree randomly
                Change the value of each variable with other randomly generated values in the variable's range and add a new tree with age 0 to the forest
    Update the best so far tree
            Sort trees according to their fitness value
            Set the age of the best tree to 0
    Return the best tree as the result.
```

In order to use the *FOA* algorithm in problem estimation, each solution is represented as shown in Figure 2.10. If a problem has *Nvar* dimensions, each tree in the representation will have *Nvar+1* variables, with the "Age" part indicating the age of the corresponding tree.

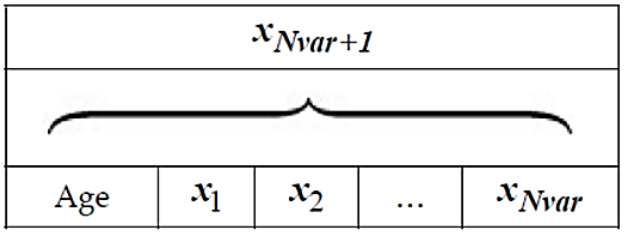


Figure 2.10 A solution representation of FOA (Ghaemi and Feizi-Derakhshi 2014)**.**

2.8 Clustering

In computer vision, image segmentation is an important aspect. The main purpose of segmentation is to obtain the resulting object in the image. Clustering is a dominant technology used to segment images. Cluster analysis involves partitioning image data sets into numerical disarticulated clusters (Aslam et al. 2020). Alternatively, clustering is putting similar image pixels into a single cluster based on some property so that the output cluster has few inter-cluster and high intra-cluster similarities. The clustering process is an unsupervised grouping or clustering of data elements (Elavarasi, Akilandeswari, and Sathiyabhama 2011). Clustering algorithms may be divided into two main categories:

Hard clustering, in which each data point is only in one cluster, like the well-known *k*-means method in Figure 2.11, while employing soft clustering, in which each data point can be part of more than one cluster, as in Gaussian mixture models. Phonemes in speech, which can be represented as a collection of different base sounds and genes, which can be involved in a variety of biological processes, are the examples in Figure 2.12 (Cluster Analysis and Clustering Algorithms - MATLAB & Simulink 2022).

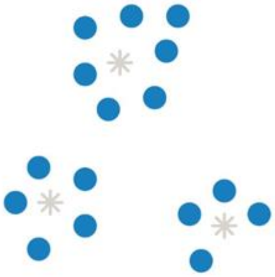


Figure 2.11 K-means clustering represents groups by their centroid - each member's average (Cluster Analysis and Clustering Algorithms - MATLAB & Simulink 2022)**.**

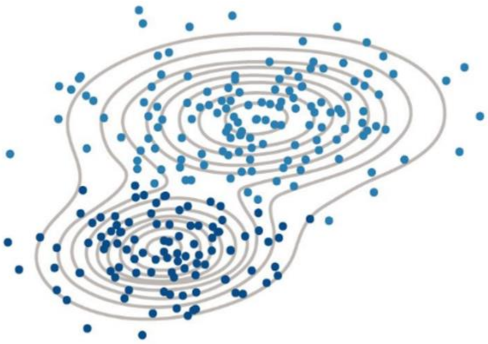


Figure 2.12 Gaussian mixture model, which assigns cluster membership probabilities, representing the strength of association with different clusters (Cluster Analysis and Clustering Algorithms - MATLAB & Simulink 2022)**.**

Many popular clustering algorithms include the *K* means, *FCM*-fuzzy *C*-Mean, Hierarchical clustering, and K-Medoids Clustering.

2.8.1 K-Means Clustering

J.B. MacQueen presented the *K*-Means algorithm, a type of cluster algorithm based on dividing (Youguo Li and Wu 2012). Data mining and pattern recognition typically make use of this unsupervised algorithm. This algorithm aims to minimize the cluster performance index and squre the error criterion. This method seeks the optimizing result by looking for *K* divisions that meet a particular requirement. First, select a point to represent the initial cluster center (usually, select the first *K* sample point to represent the initial cluster center); Secondly, we will gather the remaining sample dots to their focal points following the minimum distance criterion. If the classification is unreasonable, we will modify it (calculate each cluster focal point once more) and repeatedly iterate until we obtain a reasonable classification.

2.8.2 FCM- fuzzy C-Mean clustering

A technique for processing data that assigns a partial membership value to each image pixel is known as fuzzy logic. The fuzzy set's membership value ranges from 0 to 1. Fuzzy clustering is essentially multi-valued logic that makes it possible to use intermediate values, like when members of one fuzzy set are also members of other fuzzy sets in the same image. The distinction between full membership and non-membership is gradual. The image's fuzziness and the information it contains are both defined by the membership function. The membership function is defined by three primary fundamental characteristics: boundary, support, and core. The boundary and support determine the extent of the fuzzy set, while the core represents the full inclusion of the set. The boundary is the partial or intermediate membership between 0 and 1, while the support is the set's non-membership value (Selvakumar, Lakshmi, and Arivoli 2012).

2.9 Motion Detection

Even though a still image contains much information, cinematographers, home video fans, and video (vloggers) have been drawn to sequences of still images. The popularity of video as a medium is mostly due to its ability to capture motion; a single image offers a snapshot of the scene, whereas a succession of images also records the dynamics of the scene. The captured movement provides a robust signal for human vision, enabling us to readily identify objects as soon as they start moving, even if they are difficult to discern when stationary. There are two reasons why motion is equally important to video compression and processing. First, information about the spatiotemporal relationships between objects in a camera's field of view is carried by motion. This data can be used by traffic monitoring and security surveillance applications to figure out what is moving and who is entering and leaving a scene. Secondly, image properties such as brightness and color have very high correlations with the direction of movement, i.e., they do not change significantly with time (car colors do not change when moving along the camera field) (Konrad 2009).

The ideal goal of segmentation is to group pixels that belong to semantically meaningful parts of an image. While it is currently hard to separate static objects from moving ones in an image, using the motion information in the image makes it more practicable to separate moving ones from a dynamic scene. In image sequence processing and analysis, important functions include segmenting moving objects into an image sequence. The moving objects can be used for various purposes once detected or extracted. It has important applications in medical diagnosis and treatment, remote sensing, and video surveillance (D. Zhang and Lu 2001).

There are many algorithms used to detect motion, such as background subtraction, frame differences, and optical flow:

2.9.1 Background Subtraction Method

The background subtraction method (*BSM)* is a frequently used technique for identifying objects in a video. This algorithm compares the video's moving parts to a foreground and background image.

The purpose of this technique is to separate the foreground object from the background and then compare it to a reference frame without the object present. The disparities between the two frames will be computed, resulting in a distance matrix. Basically, the difference in the values of two frames, one without an object and the other counting an object, is compared, as well as the threshold value. The first few video frames serve as the basis for the predetermined threshold, and if the difference between the pixel values of the two frames exceeds this threshold, a moving object is detected. The threshold value differentiates between stationary background elements and moving objects.

The background subtraction technique considers that the input video frame, labeled "*I*," is composed of a static background, labeled "*B*," which remains constant throughout the scene and is in front of a moving object being observed. Because of this, every moving object is colored differently from the static background, making it possible to calculate the value difference between two frames in Algorithm 2.3.

Algorithm 2.3

```
 Input frame ($I(x,y,t)$)
 Initialize frame as background ($B(x,y,t)$)
If Difference $(I,B)$ > Threshold value Then
      Return (foreground object exists)
   Else
Return (No foreground object exits)
```

Applications like traffic monitoring, object tracking, and human action recognition systems extensively use the background subtraction method *BSM*. Determining the threshold value is crucial for the effectiveness of the background subtraction method. Various methods exist for choosing a threshold value. Automatic thresholding refers to the process of manually selecting a threshold value. Its primary drawback is the BSM's inability to adapt to a sudden shift in lighting and illumination. As a result, care should be taken when selecting the threshold parameter (Alzughaibi, Hakami, and Chaczko 2015).

2.9.2 Frame Differences

Frame differences are one of the most common methods used to detect the movement of objects. The frame difference method detects every object's motion; the camera captures that. This method is flexible in the sense that it can be modified and adjusted to match the requirements of the system. The frame difference algorithm compares every pixel within two frames sequentially, adding their differences to a block. The differential pixel values are depicted in Equation (2.32), with Δn representing the differential value on the n^{th} frame and I_n representing the pixel intensity on the n^{th} frame (Singla 2014).

$$\Delta_n = |I_n - I_{n-1}| \qquad \textbf{(2.32)}$$

By comparing the value of Δn to a threshold that has been specified, the object's motion can be calculated after the value of Δn has been obtained. Most of the time, the threshold value is within 15 percent of the range of the observed pixel intensity. Therefore, the threshold that will be utilized will be rounded up to 40 if the range falls within 0-255. The equation below can be used to determine the motion represented by (M_n):

$$M_n = \begin{cases} I_n, & \Delta_n \geq thr \\ 0, & \Delta_n \leq thr \end{cases} \qquad \textbf{(2.33)}$$

Using more than two frames to compare or an adaptive threshold value can improve this method. The frame difference method can be broken down into a few steps: first, video inputs are collected and converted into frames that will be compared; then, the algorithm creates pixel values that represent the motion detected on each frame through binarization. This binarization would produce frames in black-and-white, with white pixels representing the captured motion (Husein et al. 2019).

2.9.3 Optical Flow

Optical flow is a method used to estimate the motion of objects in an image. It is based on the idea of tracking the movement of pixels between two or more frames of an image sequence. Objects in an image can be identified using optical flow, tracking their motion, and estimating their speed and direction. It can also be used to estimate the motion of a camera, allowing for the creation of smooth transitions between frames (Kolesov et al. 2010).

2.10 Morphological Operations-Based Image Segmentation

The term "morphology" usually refers to the biological divisions of animal and plant types and structures. In mathematical morphology, the same word is used to extract the image components that are useful for describing and defining regions such as boundaries, skeletons, and convex shells. Morphological processes such as morphological filtering, thinning, and pruning are applicable for pre and post-processing (Gonzalez, Woods, and Eddins 2009).

In morphological techniques, small shapes or models called structural elements are used. Structural elements are located at every possible location in the image and are comparable to the corresponding pixel neighborhood. Structural elements are composed of small matrices of pixels and binary images, where each pixel has a value of either 0 or 1. The shape of a structural element is determined by the arrangement of its 0s and 1s, and typically, one of the pixels is designated as the origin of the structural feature. The size and center pixel is often called the pattern's origin. The size and center pixel is commonly called the structuring feature's origin (Chaki and Dey 2019).

2.10.1 Dilation and Erosion

Morphological image processing involves two important operations: dilation and erosion. Dilation involves increasing the size or thickness of objects in an image using a structuring element, which determines the specific manner and extent of thickening. The dilation of set *A* by set *B*, denoted as $A \oplus B$, is obtained by translating and reflecting set *B* and overlaying it on set A. The process of dilation is similar to spatial convolution (Gonzalez, Woods, and Eddins 2009). A sample of the dilation result is depicted in Figure 2.13.

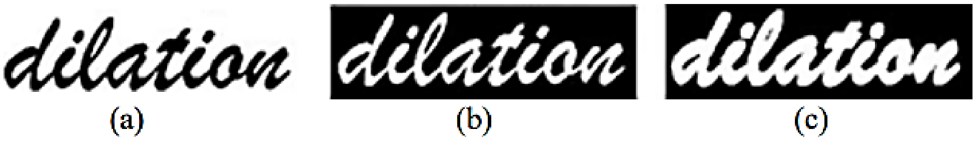


Figure 2.13 Example of using dilation (a.) original image (b.) binary image (c.) dilated image (Chaki and Dey 2019)**.**

On the other hand, erosion involves decreasing the size or thickness of objects in an image using a structuring element, which determines the specific manner and degree of shrinkage. The erosion of set *A* by set *B*, denoted as $A \ominus B$, is defined as the collection of all origin locations for structuring elements in which no part of *B* overlaps the background of *A* (Gonzalez, Woods, and Eddins 2009). An example of erosion is shown in Figure 2.14.

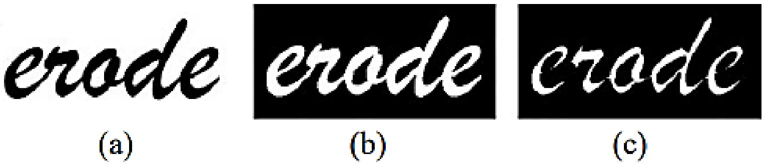


Figure 2.14 Example of using erosion (a.) original image (b.) binary image (c.) eroded image (Chaki and Dey 2019)**.**

2.10.2 Opening and Closing

Dilation and erosion are frequently used in combination in image processing applications. The image is submitted to a series of dilations and/or erosions with the same or sometimes different structural elements. The morphological opening of set *A* by set *B*, denoted as $A \circ B$, is obtained by first performing erosion on set *A* using set *B* and then dilating the result using set *B* (Gonzalez, Woods, and Eddins 2009). The opening operation creates gaps between connected regions within an image. Once the connected regions within an image are opened using a particular structuring element, further image opening using the same structuring element has no effect (Chaki and Dey 2019).

On the other hand, the morphological closing of set *A* by set *B*, denoted as $A \bullet B$, is obtained by first performing dilation on set *A* using set *B* and then erosion of the result using set *B* (Gonzalez, Woods, and Eddins 2009). The closing operation is used to fill or connect holes in image regions while preserving the initial size of the region. Once the holes are connected in the image using structural elements, further closing the image using the same structural element has no effect (Chaki and Dey 2019). Figure 2.15 illustrates the differences between opening and closing.

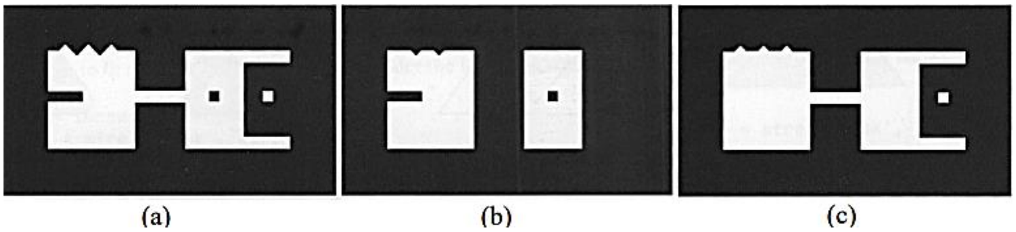


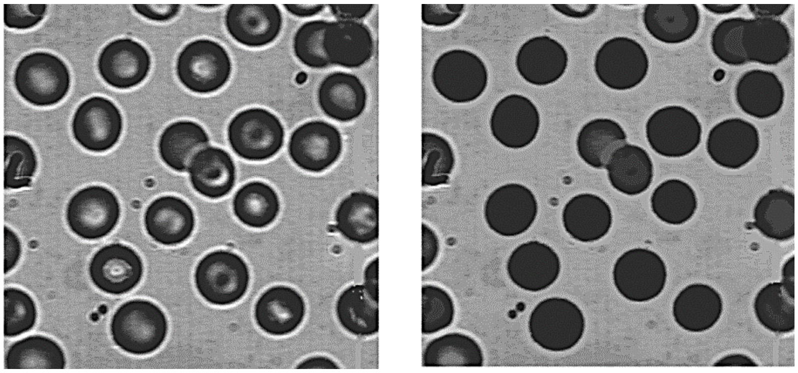
Figure 2.15 Example of opening with closing (a.) original image (b.) opening result (c.) closing result (Gonzalez, Woods, and Eddins 2009**).**

2.10.3 Filling Holes

The Morphological Reconstruction technique has numerous practical uses, including selecting marker and mask images. For example, suppose a binary image is represented by *I* and a marker image is represented by *F*. *F* will be set to 0 everywhere except on the image's border, where it is set to 1-

I (Gonzalez, Woods and. Equation (2.34) can represent this. The result is a binary image with all holes filled and equal to I, as shown in Figure 2.16.

$$F(x,y) = \begin{cases} 1 - I, & if\ f(x,y)\ is\ on\ the\ border\ of\ \ I \\ 0, & otherwise \end{cases} \quad \textbf{(2.34)}$$



(a) (b)

Figure 2.16 Example of filling effect (a.) original image (b.) filling holes result (Morphological Operations (Image Processing Toolbox) 2022)**.**

2.11 Camera Parameter Calibration

Calibrating a camera is important for many computer vision and image processing applications, such as 3D reconstruction, object tracking, and augmented reality. Accurate calibration allows for precise measurements, accurate reconstructions of 3D scenes from 2D images, and robust tracking and registration of objects in different images.

There are several methods for calibrating a camera, such as Tsai’s method (Tsai 1987) and Sturm’s method (Sturm and Maybank 1999) but one of the most widely used and convenient methods involves using a checkerboard pattern (Z. Zhang 2000) of known dimensions and orientations.

2.11.1 Checkerboard Calibration

Calibration pattern (Z. Zhang 2000) is a fundamental technique in computer vision and image processing used to estimate the intrinsic and extrinsic camera parameters. It involves using a pattern of known dimensions and orientations. There are several patterns, such as asymmetric circles grid patterns, symmetric circles grid patterns, and regular checkerboards (Calibration Patterns - MATLAB & Simulink 2023). The pattern choice depends on the specific application and the calibrated camera characteristics. However, regular checkerboards are the most commonly used pattern because they are simple to generate and provide accurate results for many applications. It consists of a grid of black and white squares arranged in a regular pattern. The size of the squares and the number of squares in each row and column can vary depending on the application and the camera resolution, as shown in Figure 2.17.

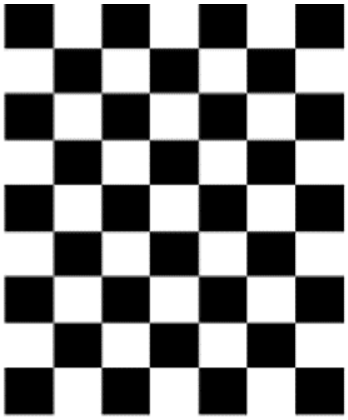


Figure 2.17 Regular checkerboard pattern

The method of detecting corners in the regular checkerboard pattern for camera calibration is typically done using the "corner detection algorithm." This algorithm works by identifying the points in the image where the gradient of the image changes sharply, indicating the presence of an edge or corner.

One commonly used corner detection algorithm is the Harris corner detector (Harris and Stephens 1998), which is based on computing the eigenvalues of the image gradient matrix. The algorithm first computes the gradient of the image using the Sobel or Scharr operator. Then, each pixel in the image computes a score based on the gradient values in the surrounding neighborhood. Pixels with high scores are considered corners.

To apply the corner detection algorithm to the regular checkerboard pattern, the algorithm looks for the corners where the black and white squares meet in the checkerboard. Once the corners are detected, the algorithm can compute the intrinsic and extrinsic camera parameters using mathematical models, such as the pinhole or perspective projection models. Estimating these parameters involves minimizing the reprojection error, which measures the difference between the observed image points and the corresponding projected points from the estimated camera parameters.

2.11.2 Pinhole Camera Model

The pinhole camera model is a mathematical representation of how light travels through a small aperture and forms an image on a flat surface. This model is used extensively in computer vision and image processing applications, including camera calibration, as shown in Figure 2.18.

The pinhole camera model has two main components: intrinsic parameters and extrinsic parameters. The intrinsic parameters describe the internal properties of the camera, such as the focal length, principal point, and distortion coefficients. The extrinsic parameters describe the position and orientation of the camera relative to the world coordinate system.

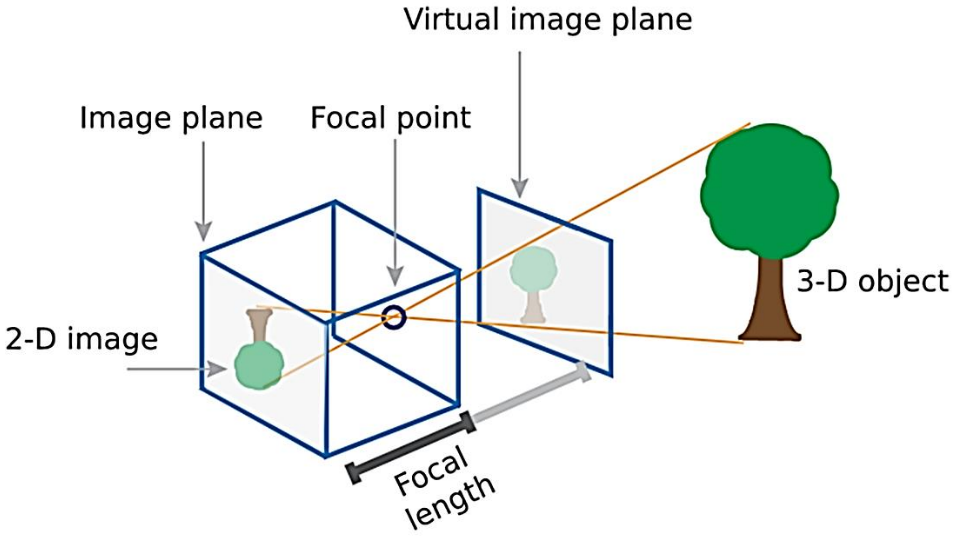


Figure 2.18 Pinhole camera model

2.12 Performance Measurements

Quality indicators concerning previous related work must be measured to determine new technologies' effectiveness and reliability.

Four elements forms are typically considered when measuring performance using confusion matrix criteria (Saponara, Elhanashi, and Gagliardi 2021):

- The true positives (*TP)* refer to the number of flame/smoke frames that were correctly identified as flame/smoke.
- The true negatives (*TN)* refer to the number of non-flame/smoke frames that were correctly identified as non-flame/smoke.
- The false positives (*FP)* refer to the number of flame/smoke frames that were incorrectly identified as non-flame/smoke.
- The false negatives (*FN)* refer to the number of non-flame/smoke frames that were incorrectly identified as flame/smoke.

Equations (2.35) – (2.37) are used to calculate performance measures. These measures include recall, precision, and accuracy, which respectively refer to the percentage of real positive frames correctly identified as positive, the ratio of true positive frames correctly detected, and the ability to discriminate between true positive and true negative pixels.

$$Recall = \frac{TP}{TP + FN} \tag{2.35}$$

$$Precision = \frac{TP}{TP + FP} \tag{2.36}$$

$$Accuracy = \frac{TP + TN}{TP + FP + TN + TP} \tag{2.37}$$

Chapter Three: The Proposed System of Fire Detection

3.1 Introduction

Fire is a significant source of danger and destruction, with the potential to cause significant harm to property, lives, and the environment. One of the main problems with fire is its ability to spread rapidly, making it challenging to contain or extinguish once it starts. Additionally, fires can be highly destructive, causing extensive damage to buildings and other structures and harm to people and animals.

Traditional methods of fire detection systems are based on smoke detectors. Although this method is reliable for buildings with low ceilings, it is inefficient for huge closed sectors with high ceilings and open areas. Recently, artificial intelligence technologies, computer vision, and image processing have been widely used in many applications; One of these applications that can employ these techniques is fire detection. So, the proposed approach in this work is based on computer vision to detect fire at the first moments of occurrence automatically. Therefore, the proposed approach is suitable and efficient for outdoor and huge indoor areas.

The proposed system of fire detection combined two approaches flame and smoke detection. The first approach detects the color of the fire, whether flame or smoke. The second approach detects the motion of fire elements such as flame and smoke. Figure 3.1 shows a general block diagram of the proposed method. First, the input video in *RGB* color space is captured from online cameras or entered from the computer's storage. Next, it is framed as separated *RGB* images and decomposed in the preprocessing stage. Then, the decomposed frames are entered into the color and motion detection stages to obtain specific fire features. After that, the processed frames for the color and motion stages are processed morphologically to remove small and unwanted objects. Finally, the area of the detected fire is computed to decide, based on the threshold, if it is a real fire or not.

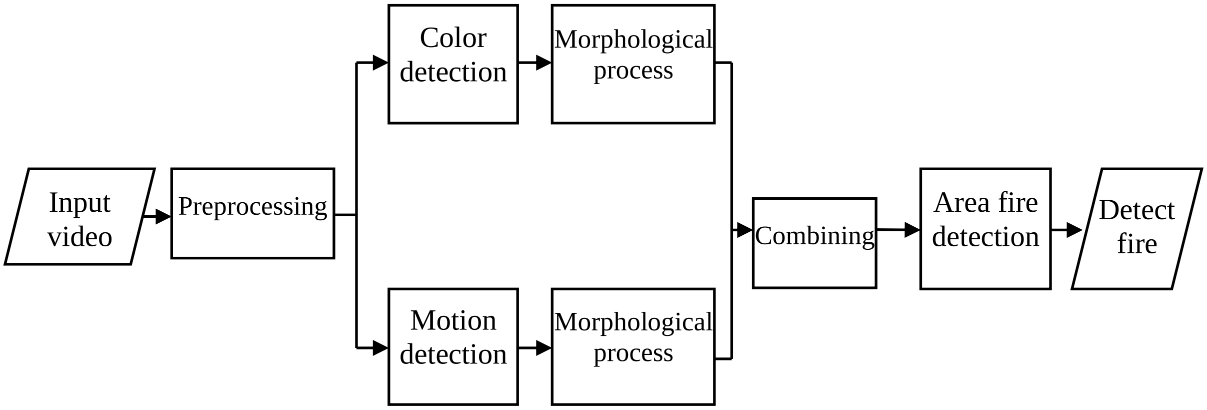


Figure 3.1 Block diagram of the proposed system

3.2 Preprocessing

The preprocessing stage is used to facilitate the next stage of operation. It includes video file entering and framing, *LWT*, and noise removal. Figure 3.2 shows the steps of the preprocessing stages. The steps of preprocessing are shown below:

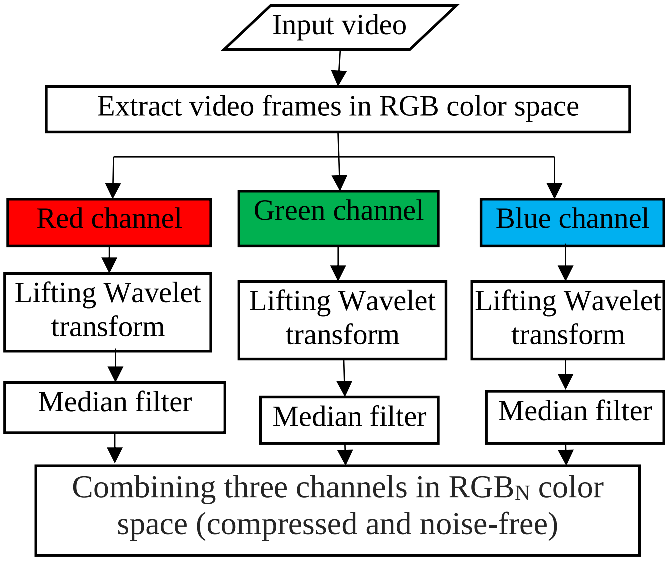

Input video
Extract video frames in RGB color space
Red channel
Green channel
Blue channel
Lifting Wavelet transform
Lifting Wavelet transform
Lifting Wavelet transform
Median filter
Median filter
Median filter
Combining three channels in RGBN color space (compressed and noise-free)

Figure 3.2 The steps of the preprocessing

3.2.1 Video File Entering and Framing

When a video file is entered into the system, it is typically broken down into individual frames from 1 to *Z*, each with a size *J*×*K*×*3*. Where *Z* represents the video length and *J, K* represent the row and column frames. Each frame represents an *RGB* image, which is further processed to extract specific information or features. One common way to process video frames is to extract the individual red, green, and blue (*RGB*) layers.

Each pixel in an image can be represented by a combination of red, green, and blue values. These values can be extracted from the video frames using the following equations:

$$Red_i = Frame_i(1{:}J, 1{:}K, 1), i = 1,2, \ldots\ldots, Z \quad \textbf{(3.1)}$$

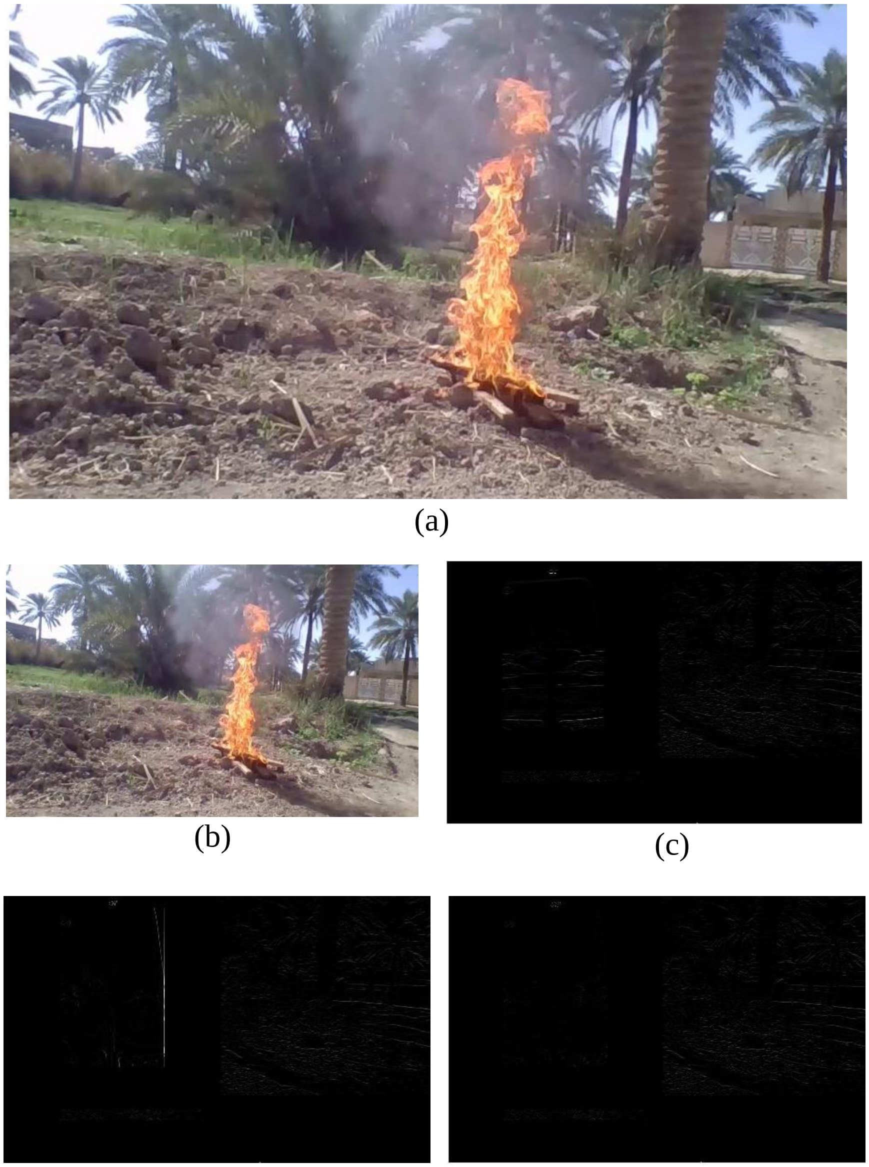
$$Green_i = Frame_i(1{:}J, 1{:}K, 2), i = 1,2, \ldots\ldots, Z \quad \textbf{(3.2)}$$

$$Blue_i = Frame_i(1{:}J, 1{:}K, 3), i = 1,2, \ldots\ldots, Z \quad \textbf{(3.3)}$$

3.2.2 Lifting Wavelet Transform

The integer lifting wavelet transform is a fast and efficient method for decomposing a signal or image into its frequency components. It is particularly useful for image and video compression, and signal processing applications. In the first stage of the preprocessing, the one-level integer Haar lifting wavelet transform (*Int-to-Int-HLWT)* is applied to each layer of the *RGB* color space in 2D form, as shown in Figure 3.2. The purpose of using the *Int-to-Int-HLWT* was to reduce the input frame size and data to 75% while saving the important data features; Figure 3.3 shows the applied *Int-to-Int-HLWT* on the original frame.

A few steps are required to be followed when using *Int-to-Int-HLWT* transformer in a proposed system; they are the following steps:



(a)

(b)

(c)

(d)

(e)

Figure 3.3 Example of the LWT process (a.) The original RGB Frame$_i$ (b.) lower band (c.) horizontal band (d.) vertical band (e.) diagonal band.

1. Each layer (*Red$_i$*, *Green$_i$*, and *Blue$_i$*) from the *Frame$_i$* is divided into blocks of size 2 × 2.
2. Employ Equation (2.15) to transform blocks based 2D *Int-to-Int-HLWT*.
3. From each transformed block, four sub-bands are obtained. The first one represents the approximation sub-band that contains the most

important information. This band contains whole features of low frequencies, as shown in Figure 3.3 (b). The remaining sub-bands, which include the horizontal, vertical, and diagonal detail sub-bands in Figure 3.3 (c, d, and e), contain high-frequency information and represent the fine details in the image.

4. The resulting approximation sub-bands are combined to reconstruct the compressed new layers to form the new *RGB* frame with a size of 25% of the original size without losing any important feature of the original frame.

3.2.3 Median Filter

The median filter is a common digital non-linear filter for removing noise from an image. It replaces each pixel in the image with the median value of the pixels in a neighborhood around it. In the second preprocessing stage, the median filter removes noise or eliminates a single or few pixels. To use the median filter in the proposed system, the following steps are followed:

1. Each layer (Red_i, $Green_i$, and $Blue_i$) from $Frame_i$ has "salt and pepper" noise that added to demonstrate median filter processing.
2. Define the neighborhood size around each pixel that will be used to calculate the median value. The suitable window size is 3 × 3.
3. Iterate through each pixel in the layer and calculate the median value of the pixels within the defined neighborhood for that pixel.
4. Replace the original value of each pixel with the calculated median value.
5. Repeat the process for all pixels in the layer.

6. The filtered $Frame_i$ in the new RBG_N color space can be used for further analysis or processing. Figure 2.3 shows the removal of noise by using the median filter.

3.3 Fire Color Detection

This work proposes two approaches for detecting the color of the flame depending on the type of threshold: the first is characterized by the ability to detect the color of the flame in outdoor locations. In contrast, the second is an adaptive method for different locations (indoor and outdoor). For smoke detection, a multi-threshold method is suggested for detecting different colors of smoke.

3.3.1 Smoke Color Detection

Smoke is characterized by its color, which can vary from a whiteish gray to a blackish gray. Despite the wide range of colors within the smoke class, color remains a prominent feature for smoke detection. *RGB* frames are commonly used for this purpose, as is seen in Figure 3.5. However, one drawback of using this color space for detecting flames and smoke is that the sky and flame pixels may appear oversaturated in the *R* and *G* channels due to the lack of separation between luminance and chrominance, as is seen in Figure 3.4. International Commission on Illumination *CIE L*a*b** color space transformation is used to solve this problem in a new and better way.

To detect the color of smoke in a proposed system, these steps have been performed as shown in Figure 3.5:

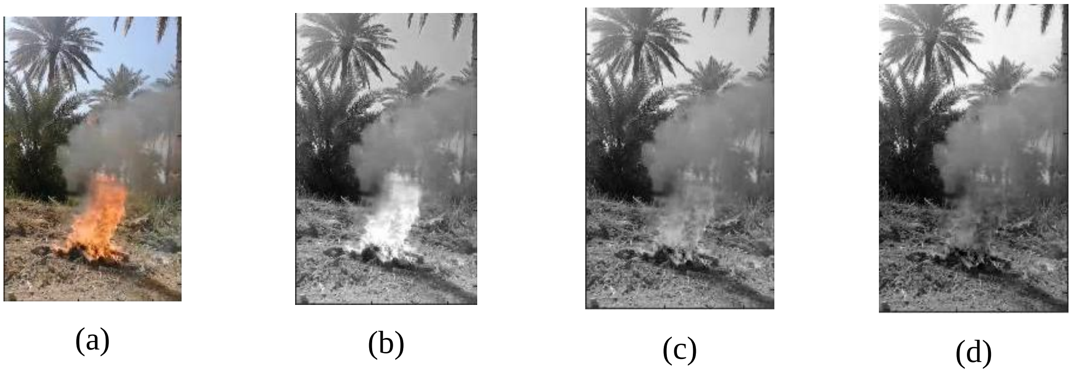


Figure 3.4 The flame and smoke RGB$_N$ frame and splitted R-G-B layers at outdoors. (a.) ogrinal RGB$_N$ frame (b) R layer. (c) G layer. (d) B layer.

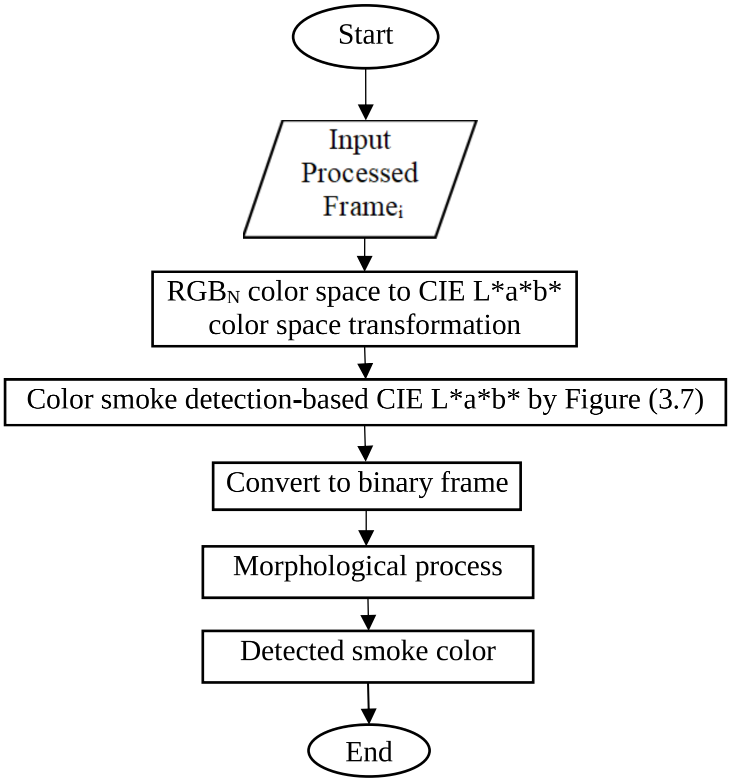
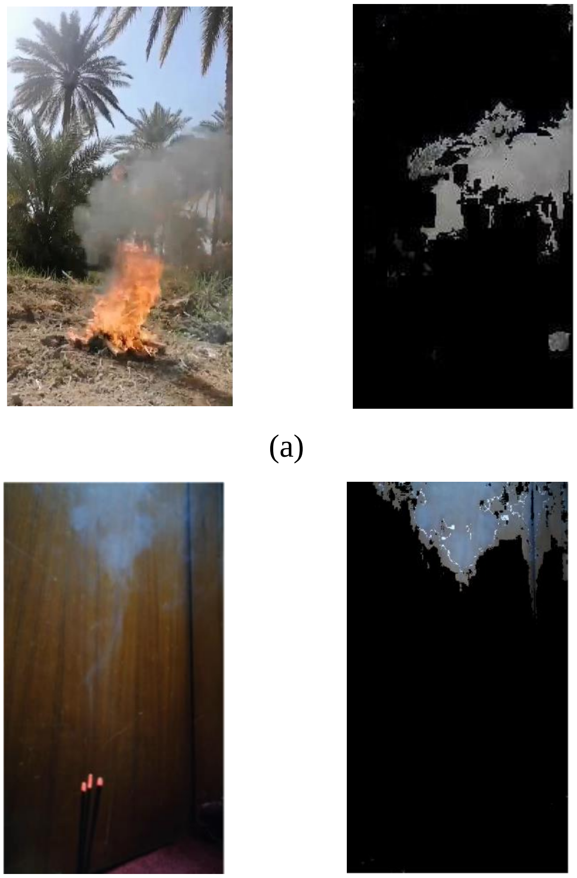

Start
Input Processed Frame$_i$
RGB$_N$ color space to CIE L*a*b* color space transformation
Color smoke detection-based CIE L*a*b* by Figure (3.7)
Convert to binary frame
Morphological process
Detected smoke color
End

Figure 3.5 Flowchart of the smoke detection based color.

1. The color space transformation from *RGB$_N$* color space to *CIE L*a*b** color space and the transformation equations are written in Equations (2.7) and (2.8).
2. Use the *CIE L*a*b** color space's multi-threshold to segment the frame. The result is the region of the smoke-based color (*R*eg$_{color}$) in *CIE*

*L*a*b** color space, which represents the foreground, while the background represents non-color smoke, as shown in Figure 3.6 for indoor and outdoor places. Then, the multi-thresholds are applied for each frame according to the following Equations:



(a)

(b)

Figure 3.6 Smoke detection based on CIE L*a*b*. (a.) RGB$_N$ and detected smoke frames for outdoor (b) RGB$_N$ and detected smoke frames for indoor

$$thr_{low_j} \leq L_{mask_j} \leq thr_{high_j} \qquad j = 1,2,3$$
$$thr_{low_j} \leq a_{mask_j} \leq thr_{high_j} \qquad j = 1,2,3 \qquad \textbf{(3.4)}$$
$$thr_{low_j} \leq b_{mask_j} \leq thr_{high_j} \qquad j = 1,2,3$$

$$Lab_{mask_j} = L_{mask_j} \times a_{mask_j} \times b_{mask_j} \qquad j = 1,2,3 \qquad \textbf{(3.5)}$$

$$R_{New_j} = Lab_{mask_j} \times Red \qquad j = 1,2,3 \qquad \textbf{(3.6)}$$

$$G_{New_j} = Lab_{mask_i} \times Green \qquad j = 1,2,3$$

$$B_{New_j} = Lab_{mask_j} \times Blue \qquad j = 1,2,3$$

$$RGB_{New\,j} = combin\,(R_{New_j},\ G_{New_j}, B_{New_j})$$

$$RGB_{New} = \sum RGB_{New_j} \qquad j = 1,2,3 \tag{3.7}$$

where thr_{low} and thr_{high} represent the low and high thresholds that give the binary mask and are used to select certain values from the original data. *New* represents the new value of *R*, *G*, and *B*, where some value was selected from the overall value after applying the binary mask.

Table 3.1 includes the value of the multi-thresholds (static thresholds) of the *CIE L*a*b** depended on the trial-and-error method. These thresholds represent the smoke color gradation from whitish gray to blackish gray. The steps of the smoke color detection-based *CIE L*a*b** color space are shown in Figure 3.7.

Table 3.1 The multi-thresholds of CIE L*a*b* to detect smoke colors

Thresholds The rules	First thresholds	Second threshold	Third threshold
$\boldsymbol{Rule\ 1}$: L	$\begin{pmatrix}\text{Low: } 0.058\\ \text{High: } 98.32\end{pmatrix}$	$\begin{pmatrix}\text{Low: } 87.955\\ \text{High: } 100\end{pmatrix}$	$\begin{pmatrix}\text{Low: } 8.876\\ \text{High: } 63.748\end{pmatrix}$
$\boldsymbol{Rule\ 2}$: a	$\begin{pmatrix}\text{Low: } -8.067\\ \text{High: } 4.853\end{pmatrix}$	$\begin{pmatrix}\text{low: } -2.211\\ \text{high: } 5.546\end{pmatrix}$	$\begin{pmatrix}\text{Low: } -2.377\\ \text{High: } 2.773\end{pmatrix}$
$\boldsymbol{Rule\ 3}$: b	$\begin{pmatrix}\text{Low: } -24.653\\ \text{High: } -9.788\end{pmatrix}$	$\begin{pmatrix}\text{Low: } -9.359\\ \text{High: } 3.12\end{pmatrix}$	$\begin{pmatrix}\text{Low: } -10.399\\ \text{High: } 4.065\end{pmatrix}$

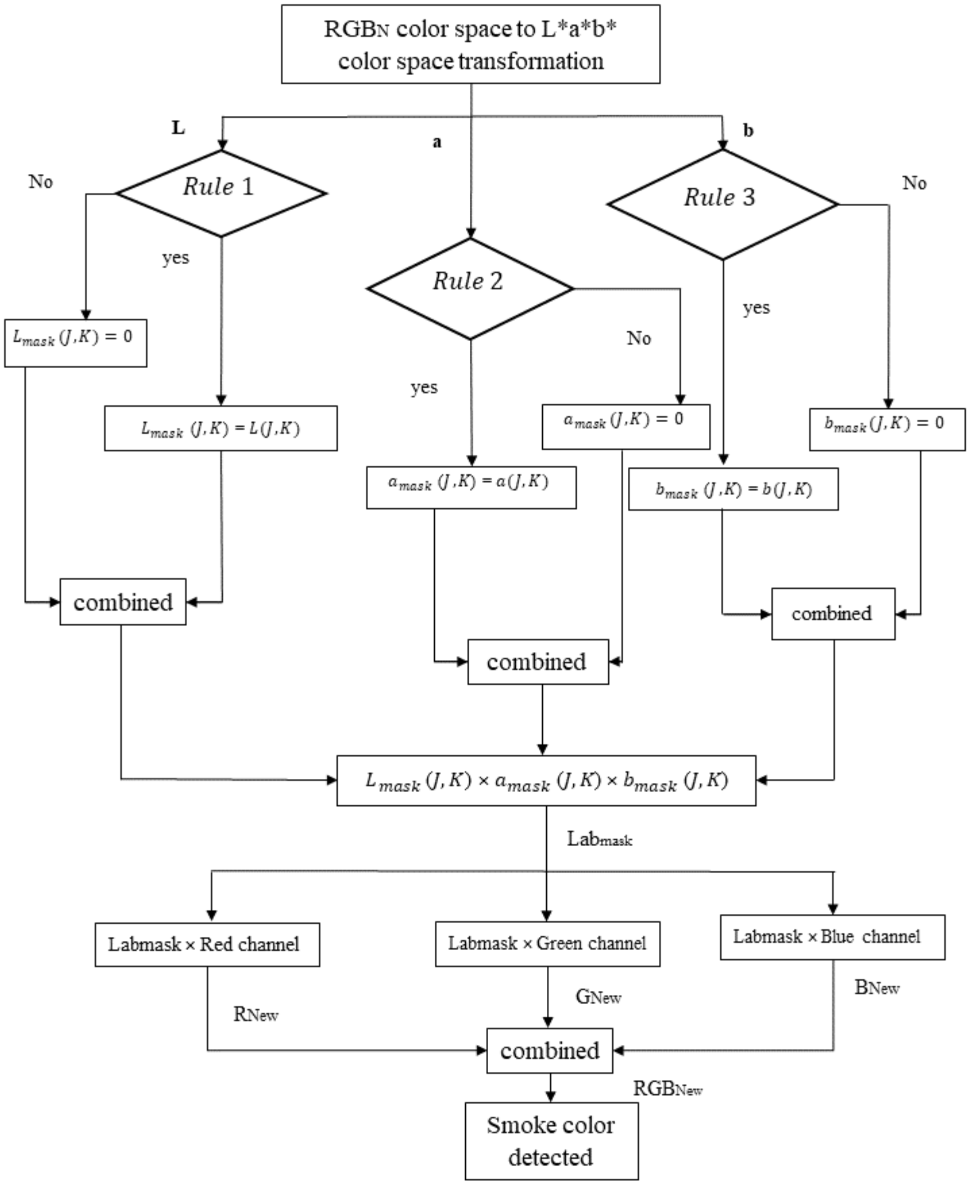
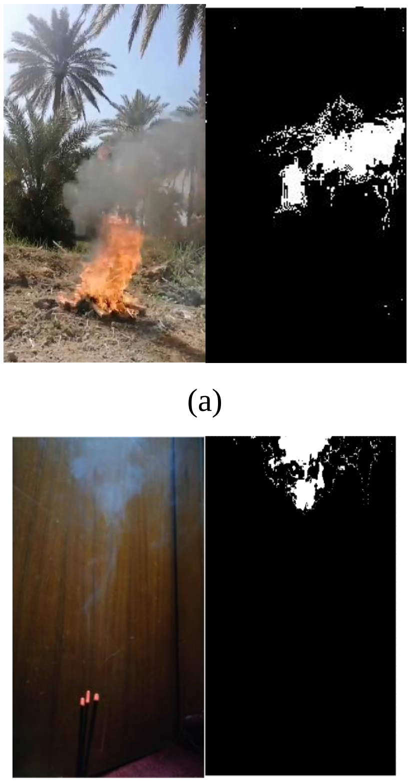


Figure 3.7 The smoke color detection based on CIE L*a*b* color space

3. Convert the frame to binary to qualify for the morphological processes and combine it with motion detection. The bitwise AND operation is used to merge the results of these two detection processes. It operates on the corresponding bits of two binary numbers. Compares each bit of the first operand to the corresponding bit of the second operand. If both

bits are 1, the corresponding result bit is set to 1. Otherwise, the corresponding result bit is set to 0.

Figure 3.8 shows the binary frames in which the white pixels represent the smoke color or a color similar to the smoke color.



(a)

(b)

Figure 3.8 Binary frame represents the smoke color (a.) For outdoor place. (b.) For indoor place.

3.3.2 Flame Color Detection based on Non-Adaptive Thresholds

Fire is characterized by its color, which can change depending on the material burned. The *RGB* color model has less computational complexity than other color models, but the *HSV* and *YCbCr* color models are often preferred due to their ability to provide a more human-centric way of describing colors. The proposed method for flame color detection integrates the *HSV* and *YCbCr* color spaces to determine potential fire zones using a static or non-adaptive threshold. The integration of the two colors increases the accuracy of fire color detection. In outdoor places, the static threshold is used to detect the color of the flame because it requires less complexity and, thus, less processing time. To detect the color of the flame in the proposed system, these steps have been performed as shown in Figure 3.9:

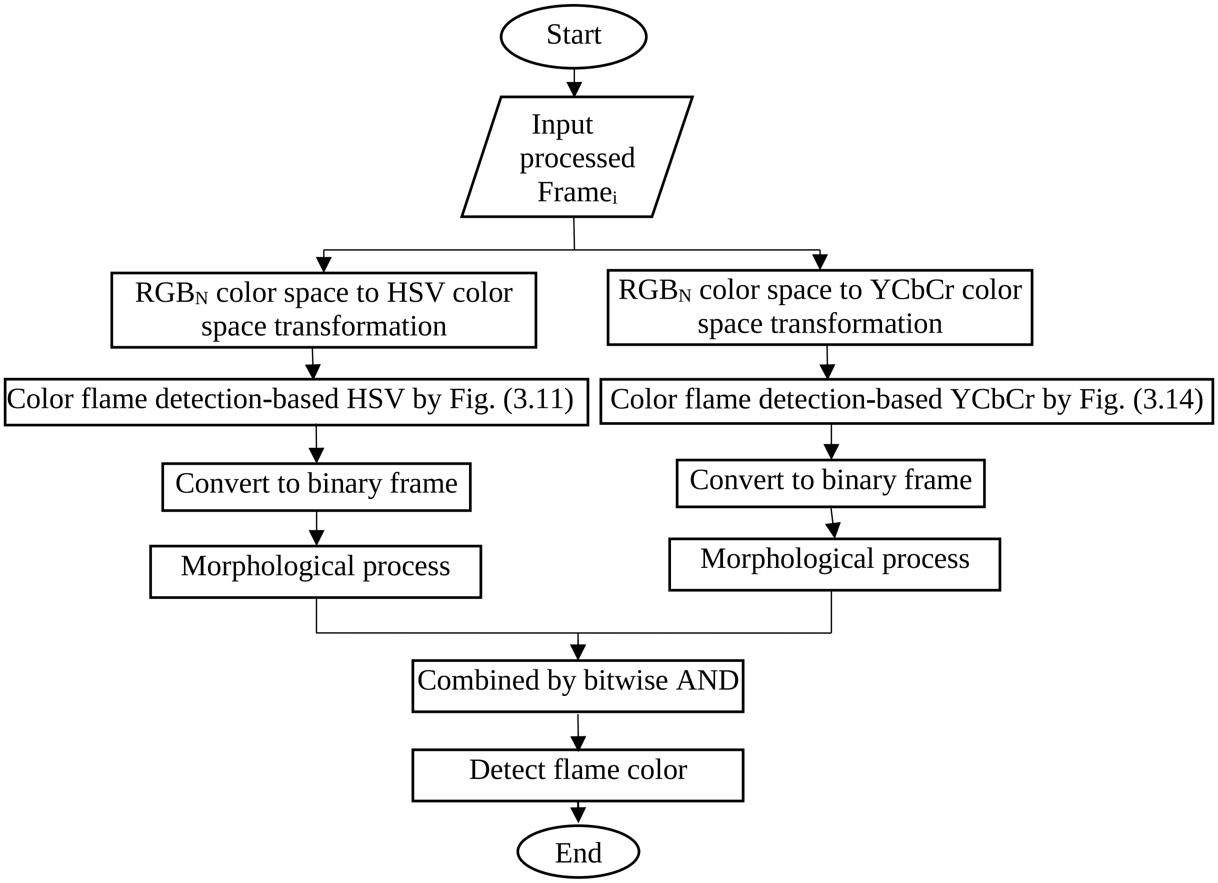
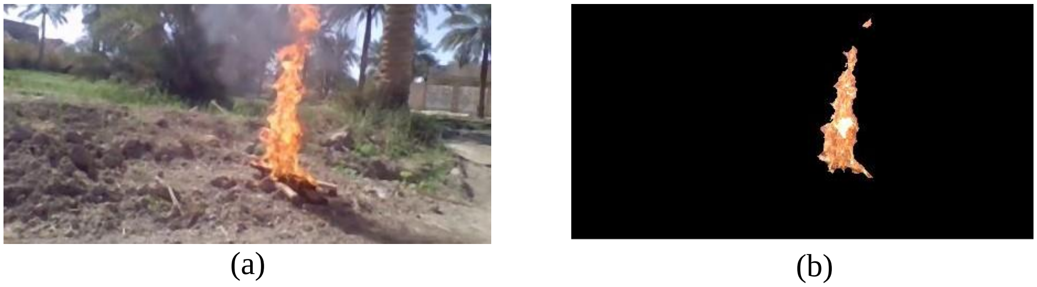

Start
Input processed $Frame_i$
RGB_N color space to HSV color space transformation
RGB_N color space to YCbCr color space transformation
Color flame detection-based HSV by Fig. (3.11)
Color flame detection-based YCbCr by Fig. (3.14)
Convert to binary frame
Convert to binary frame
Morphological process
Morphological process
Combined by bitwise AND
Detect flame color
End

Figure 3.9 Flowchart of the non-adaptive flame color detection



(a)

(b)

Figure 3.10 Flame detection based on HSV (a.) RGB_N for outdoor (b.) detected flame.

$$\begin{gathered} thr_{low} \leq H_{mask} \leq thr_{high} \\ thr_{low} \leq S_{mask} \leq thr_{high} \\ thr_{low} \leq V_{mask} \leq thr_{high} \end{gathered} \tag{3.8}$$

$$HSV_{mask} = H_{mask} \times S_{mask} \times V_{mask} \tag{3.9}$$

$$\begin{gathered} R_{New} = HSV_{mask} \times Red \\ G_{New} = HSV_{mask} \times Green \\ B_{New} = HSV_{mask} \times Blue \\ RGB_{New} = combin\,(R_{New}, G_{New}, B_{New}) \end{gathered} \tag{3.10}$$

where thr_{low} and thr_{high} represent the low and high thresholds that give the binary mask and are used to select certain values from the original data. *New* represents the new value of *R*, *G*, and *B*, where some value was selected from the overall value after applying the binary mask. The values of the thresholds are written in Table 3.2 depended on the trial-and-error method:

Table 3.2 The threshold of HSV to detect flame color

Threshold The rules	thr_{low}	thr_{high}
Rule 1: H	0	0.2
Rule 2: S	0.47	0.98
Rule 3: V	0.7	0.98

The steps for flame color detection in the *HSV* color space are illustrated in Figure 3.11.

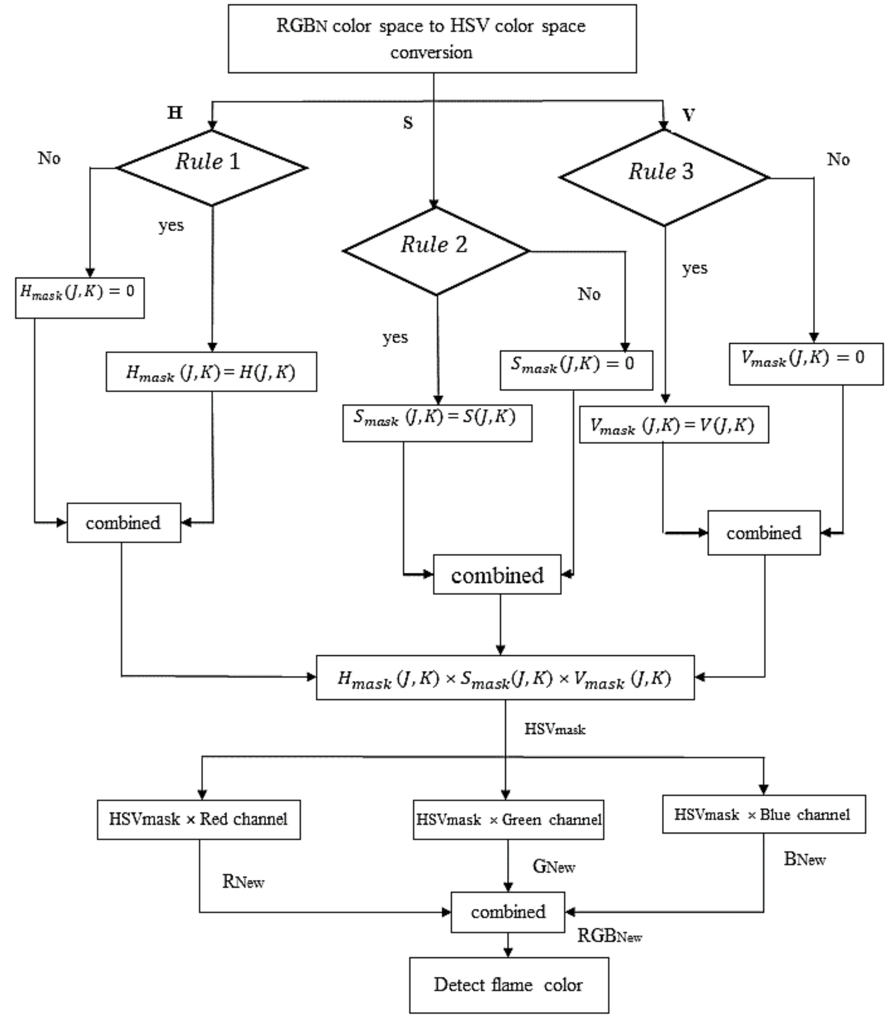
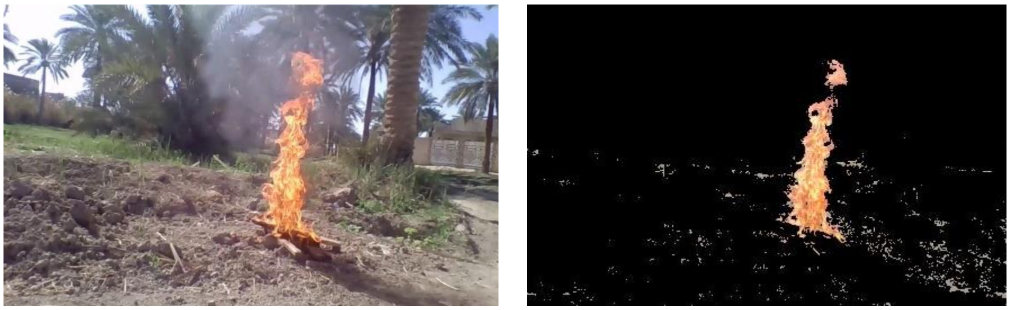


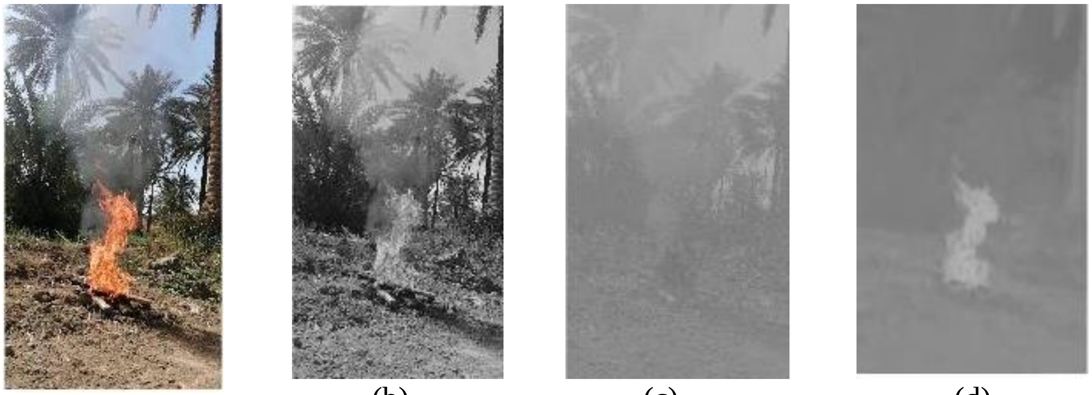
Figure 3.11 The flame color detection based on HSV color space.

3. Segment the frame using the *YCbCr* color space equations; the color of flames can be represented in the *YCbCr* color space, with the foreground representing the color of the flames and the background representing non-flame colors, as shown in Figure 3.12. Since the brightness of the flame is greater than chrominance-blue, and chrominance-red is larger than chrominance-blue, this can easily be observed from the frames shown in Figure 3.13 for flame regions. Therefore, rule one can be formulated as follows:



(a) (b)

Figure 3.12 Flame detection based on YCbCr (a.) RGB$_N$ for outdoor (b) detected flame.



(a) (b) (c) (d)

Figure 3.13 The flame and smoke RGB$_N$ frame and splitted Y-Cb-Cr layers at outdoors. (a.) original RGB$_N$ frame. (b.) Y layer. (c.) Cb layer. (d.) Cr layer

$Rule\ 1:$

$$F(J,K) = \begin{cases} 1, if\ Y(J,K) > \ Cb(J,K)\ and\ \ Cr(J,K) > Cb(J,K) \\ \ \ 0, otherwaise \end{cases} \quad \textbf{(3.11)}$$

In addition to Equation (3.11), the average values of the Y, *Cb*, and *Cr* components in the overall frame, Y-mean, *Cb*-mean, and *Cr*-mean, also contain valuable information. The flame region is often the brightest area in the image. Specifically, the value of the Y component in the flame region is greater than the average Y component of the entire frame. In comparison, the value of the *Cb* component is typically less than the average *Cb* value of the overall frame. Additionally, the *Cr* component in the flame region is larger than the average *Cr* component (Çelik and Demirel 2009), which can be summarized in the following rule:

$Rule\ 2$:

$$F(J,K) = \begin{cases} 1, if\ Y(J,K) > Y_{mean}\ and\ Cb(J,K) < Cb_{mean}\ and\ Cr(J,K) > Cr_{mean} \\ 0, otherwaise \end{cases} \quad \textbf{(3.12)}$$

$$Y_{mean} = \frac{1}{q}\sum_{ii=1}^{q} Y(J_{ii}, K_{ii}),\ Cb_{mean} = \frac{1}{q}\sum_{ii=1}^{q} Cb(J_{ii}, K_{ii})$$
$$\text{and } Cr_{mean} = \frac{1}{q}\sum_{ii=1}^{q} Cr(J_{ii}, K_{ii}) \quad \textbf{(3.13)}$$

The coordinates represent the spatial position of a pixel (J_{ii}, K_{ii}), and the mean values of the luminance (*Y-mean*) and chrominance (*Cb-mean*, *Cr-mean*) are also given, with *q* being the total number of pixels in the frame. The number of pixels in frame denoted *ii*. The region of the flame-based color ($R\mathrm{eg}_{\mathrm{color}}$), which corresponds to the pixel indices identified by the *YCbCr* color space, can be obtained by combining rules 1 and 2 in the following manner:

$$Rule\ 3 = Rule\ 1 \cap Rule\ 2 \quad \textbf{(3.14)}$$

Figure 3.14 illustrates the steps for pixel indices identified by the *YCbCr* color space. *N* represents the new values of *YCbCr* color space obtained by applying the rules to the overall values.

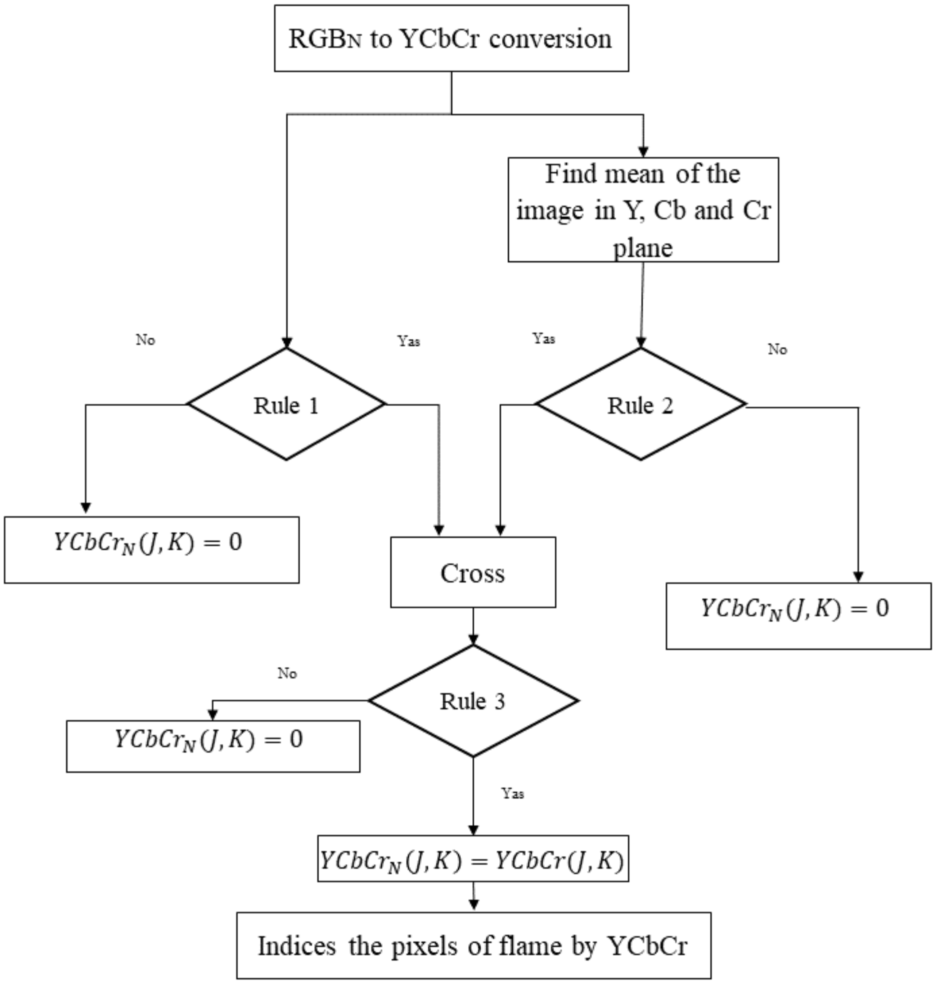


Figure 3.14 The flame pixel indices identified by the YCbCr color space.

Flame color detection after using the rules of *YCbCr* color space involves the following steps:

a. Identify the indices of the pixels that satisfy each rule 3.

b. Create a new zero frame owns the same size as the original RBG_N frame.

c. The pixels that satisfy the rules are copied from the original frame RBG_N to the new zero frame.

d. The resulting frame only shows the flame regions.

4. Convert frames of the flame color detection based on *HSV* and *YCbCr* color spaces to binary in order to prepare them for morphological processing and combining. As shown in Figure 3.15, the resulting binary frames have white pixels representing the flame color or similar colors to the flame.

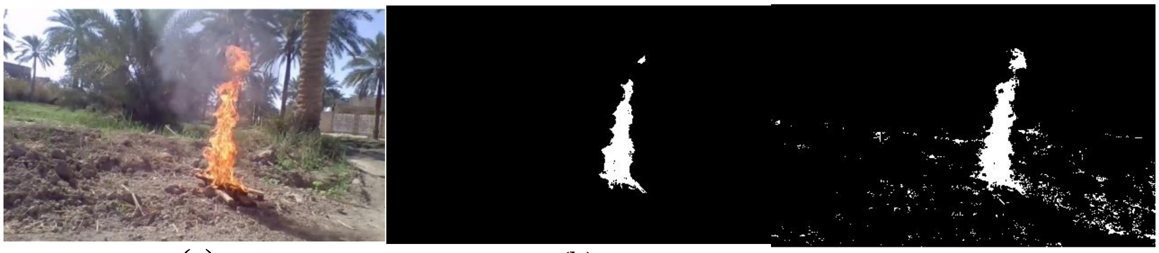

(a) (b) (c)

Figure 3.15 Binary frame represents the flame color. (a.) orginal frame. (b.) HSV; (c.) YCbCr.

3.3.3 Flame Color Detection based on Adaptive Thresholds

The second method for detecting flame colors is similar to the first method regarding work steps. Still, it uses different methods to calculate the threshold (Farshi, Demirci, and Feizi-Derakhshi 2018), as shown in Figure 3.16. This system is designed to be adaptable to different light intensities to distinguish fire from similar objects in any indoor or outdoor environment. This is achieved through multi-threshold estimation methods such as Otsu’s and Kapur’s criteria objective functions-based particle swarm optimization (*PSO*) and forest optimization algorithm (*FOA*). To detect the color of the flame in the proposed system, these steps have been performed as shown in Figure 3.16:

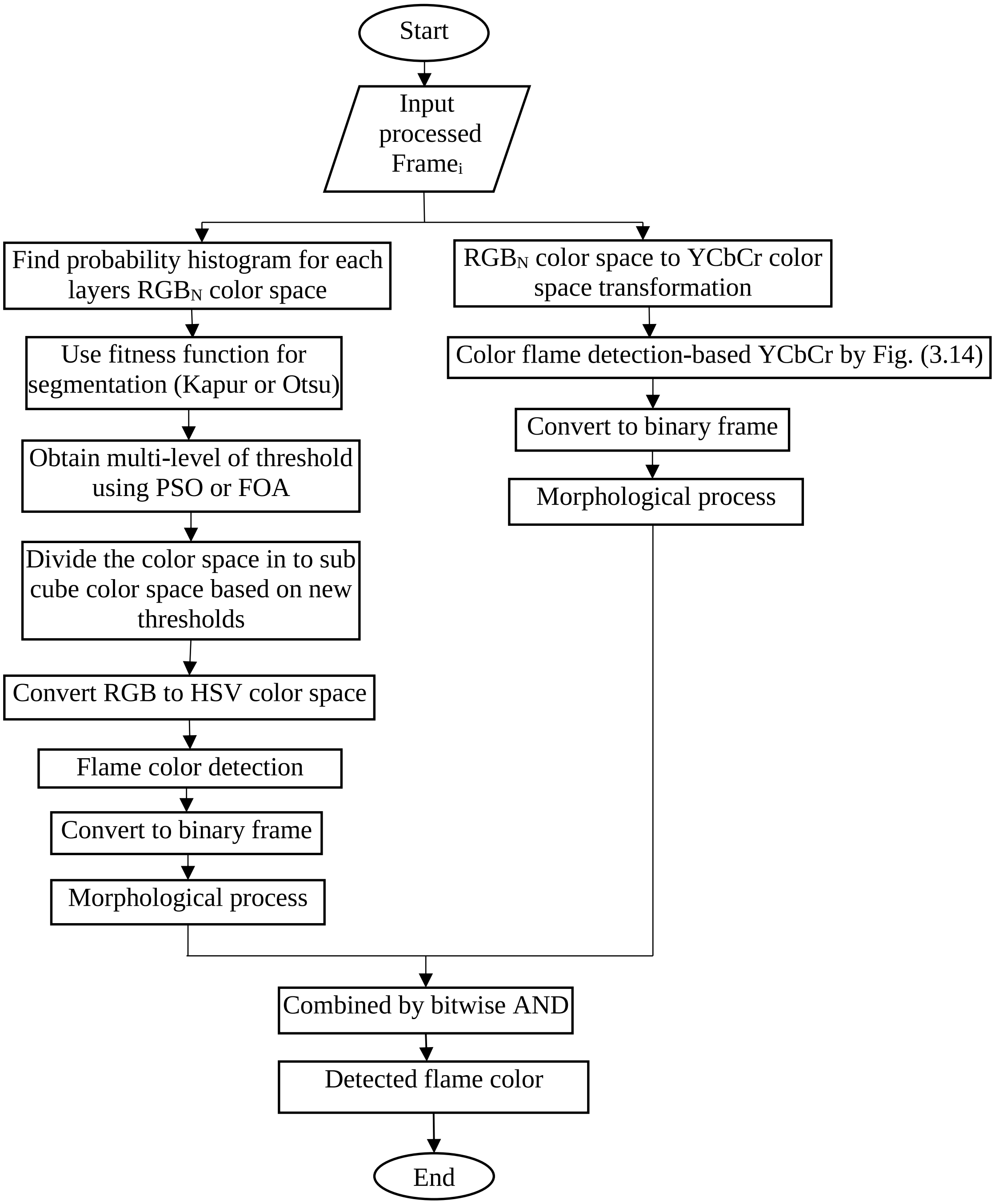

Start
Input processed Frame$_i$
Find probability histogram for each layers RGB$_N$ color space
Use fitness function for segmentation (Kapur or Otsu)
Obtain multi-level of threshold using PSO or FOA
Divide the color space in to sub cube color space based on new thresholds
Convert RGB to HSV color space
Flame color detection
Convert to binary frame
Morphological process
RGB$_N$ color space to YCbCr color space transformation
Color flame detection-based YCbCr by Fig. (3.14)
Convert to binary frame
Morphological process
Combined by bitwise AND
Detected flame color
End

Figure 3.16 Flowchart of the adaptive flame color detection

1. Find the probability histogram for each layer (*Red$_i$*, *Green$_i$*, and *Blue$_i$*) for the Frame$_i$ of each gray level from 0 to *GG*-1. The probability of a specific gray level (*e*) can be defined as:

$$p_e = {}^{h_e}/_{(J \times K)} \quad \textbf{(3.15)}$$

where J and K are the dimensions of the frame, and h_e is the number of pixels with gray level e, $0 \le e \le (GG - 1)$.

2. Otsu's between-class variance technique and Kapur's entropy criterion approach are the thresholding functions that are most frequently chosen. Maximizing the objective function extends Kapur's entropy criterion for multilevel thresholding. The histogram's partial probabilities range from w_0 to w_m, and the histogram's partial entropies range from H_0 to H_m. The number of thresholds is represented by m.

$$U(thr_1, thr_2, \dots, thr_m) = H_0 + H_1 + H_2 + \cdots + H_m \quad \textbf{(3.16)}$$

where $H_0 = -\sum_{e=0}^{t_1-1} \frac{p_e}{w_0} \ln \frac{p_e}{w_0}, \quad w_0 = \sum_{e=0}^{t_1-1} p_e$

$$H_1 = -\sum_{e=t_1}^{t_2-1} \frac{p_e}{w_1} \ln \frac{p_e}{w_1}, \quad w_1 = \sum_{e=t_1}^{t_2-1} p_e$$

$$H_2 = -\sum_{e=t_2}^{t_3-1} \frac{p_e}{w_2} \ln \frac{p_e}{w_2}, \quad w_2 = \sum_{e=t_2}^{t_3-1} p_e$$

$$\vdots$$

$$H_m = -\sum_{e=t_m}^{GG-1} \frac{p_e}{w_m} \ln \frac{p_e}{w_m}, \quad w_m = \sum_{e=t_m}^{GG-1} p_e$$

3. Otsu's between-class variance algorithm can also estimate the multilevel thresholds for any grayscale image. As a result, the following is a description of an objective function with multi-levels thresholding:

$$U(thr_1, thr_2, \dots, thr_m) = \sigma_0 + \sigma_1 + \sigma_2 + \cdots + \sigma_m \quad \textbf{(3.17)}$$

where $\sigma_0 = w_0(\mu_0 - \mu_T)^2$

$$\sigma_1 = w_1(\mu_1 - \mu_T)^2$$

$$\sigma_2 = w_2(\mu_2 - \mu_T)^2$$

$$\vdots$$

$$\sigma_m = w_m(\mu_m - \mu_T)^2$$

with

$$w_0 = \sum_{e=0}^{t_1-1} p_e, \quad \mu_0 = \sum_{e=0}^{t_1-1} \frac{ep_e}{w_0}$$

$$w_1 = \sum_{e=t_1}^{t_2-1} p_e, \quad \mu_1 = \sum_{e=t_1}^{t_2-1} \frac{ep_e}{w_1}$$

$$\vdots$$

$$w_m = \sum_{e=t_m}^{GG-1} p_e, \quad \mu_m = \sum_{e=t_m}^{GG-1} \frac{ep_e}{w_m}$$

and

$$\mu_T = \sum_{e=0}^{GG-1} ep_e$$

Both the means of the classes μ_0 to μ_m and the mean of the entire image μ_T have constraints, where $thr_1 < thr_2 < thr_3 < \ldots < thr_m$.

4. The most common and most recent algorithms have been used even though there have been numerous optimization methods. The first algorithm is *PSO*, which is widely used and inspired by the social behavior of birds and fish. The *PSO* method randomly initializes the locations of particles in the swarm, and all particles are updated according to a specific set of rules in each iteration. Equations (2.43) and (2.44) show the rules of *PSO*, while Table 3.3 includes the values of the parameters for *PSO* used in the proposed system. These values selected based on the trial-and-error method.

Table 3.3 The used parameters of PSO.

Parameters	Values
Size of the swarm " Number of particles."	N_p = 240
Maximum Iteration Value	T-max = 100
Upper band	Ub = 255
Lower band	Lb = 1
POS momentum or inertia	W = 0.98
PSO parameter R1	R1 = Random [0,1]

PSO parameter R2	R2 = Random [0,1]
PSO parameter C1: individual cognitive	C1 = 1.87
PSO parameter C2: social cognitive	C2 = 1.87

5. *FOA* is the second method used for finding multi-level thresholds. The concept behind it draws inspiration from the survival mechanisms of trees in a forest. Like some trees that persist for several decades and propagate over time, while others have a shorter lifespan, *FOA* simulates the natural method of seed dispersal to recognize exceptional trees in a forest that have germinated in the most advantageous geographical locations. The steps of *FOA* are explained in Algorithm 2.2, and the values of the parameters are listed in Table 3.4 for the proposed system. These values selected based on the trial-and-error method.

Table 3.4 The used parameters of FOA.

Parameters	Values
Maximum number of iterations	T-max = 200
Local seeding changes	LSC =1
Global seeding change	GSC =1
Upper band	Ub = 255
Lower band	Lb = 1
The limitation of the area forest	area_limit = 200
The maximum allowed Age of a tree	Life_time = 15
The percentage of the candidate population for global seeding	Transfer_rate = 10

Image histogram segmentation uses the optimization algorithms listed to find the best threshold values. The process is as follows:

a. Set the minimum and maximum threshold limits using the optimization algorithm's lower and upper bounds.

b. Initialize *m*-dimensional positions for each member of the population, with dimension values indicating the number of thresholds in the order $thr_1 < thr_2 < thr_3 < ... < thr_m$.

c. Evaluate the objective function of each member of the population using Equations (3.16) and (3.17).

d. Update the threshold values for the individual member.

e. If the termination conditions are met, the process ends; otherwise, return to step c.

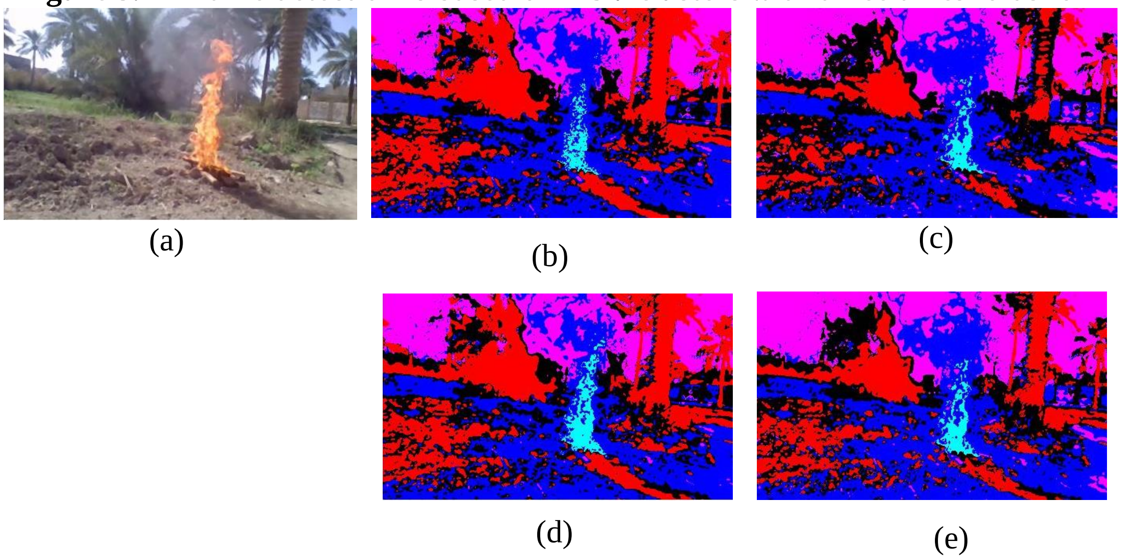
f. The optimal threshold values are the ones that correspond to the global best member.

6. The threshold values calculated for each channel produce subsets of color space by dividing the color cube RGB_N into smaller cubes or prisms. Each pixel in one of the sub-cubes or prisms is part of the same cluster. The thresholds of each channel determine the volume of each smaller cube or prism, which is unique to each image. As a result, the image's pixel intensity distributions are linked to the shapes of sub-cubes and prisms. Larger volume clusters have less homogeneity. So, increasing thresholds to increase cluster homogeneity solves this problem. Therefore, the max number of clusters that may be produced for a color image is:

$$c_m = (m + 1)^3 \qquad \textbf{(3.18)}$$

Eight sub-clusters are created when one threshold is applied to each channel *(m = 1)*. Increasing the threshold to *(m = 2)* results in 27 sub-clusters and *(m = 3)* in 64 sub-clusters. This pattern continues as the threshold value increases.

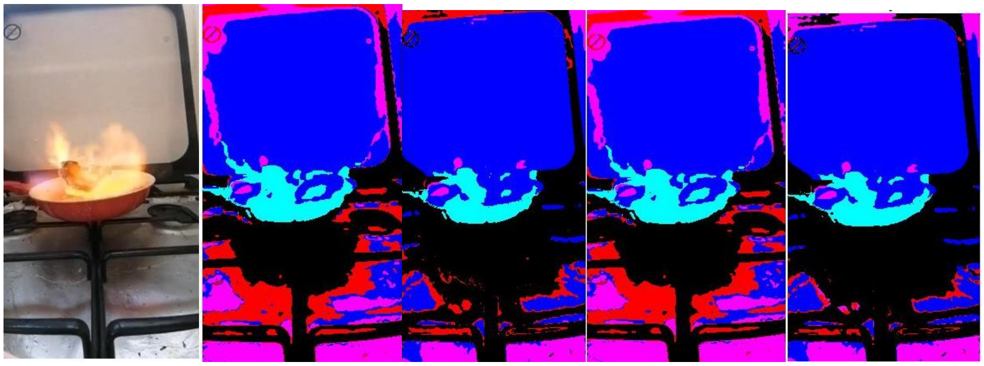
7. The color space transformation from RGB_N color space to *HSV* color space is used, for which the transformation equations are written in Equations (2.3), (2.4), and (2.5). Figures 3.17 and 3.18 illustrate the application of three thresholds to each layer in the transformation to the *HSV* color space for all cases, including Otsu, Kapur, *POS*, and *FOA* for indoor and outdoor places. The region of the flame-based color (Reg_{color}) is shown in blue sky color.



(a) (b) (c)

(d) (e)

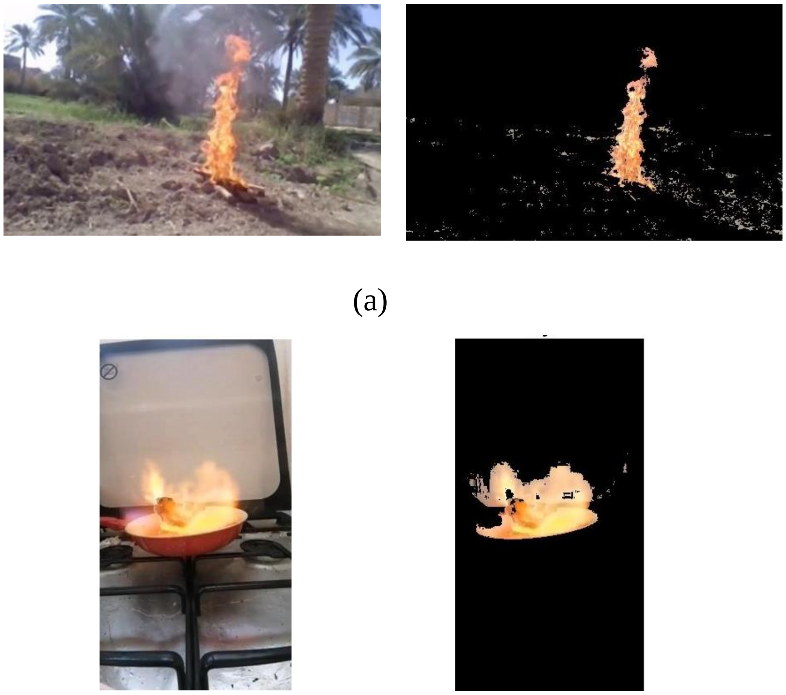
Figure 3.17 Flame detection is based on HSV clusters with three thresholds for outdoors. (a.) Frames₁ in RGBN (b.) based on POS-Otsu; (c.) based on POS-Kapur; (d.) based on FOA-Otsu; (d.) based on FOA-Kapur.



(a) (b) (c) (d) (e)

Figure 3.18 Flame detection is based on HSV clusters with three thresholds for indoors. (a.) Frames$_1$ in RGB$_N$ (b.) based on POS-Otsu; (c.) based on POS-Kapur; (d.) based on FOA-Otsu; (e.) based on FOA-Kapur.

8. Segment the frame using the *YCbCr* color space equations; all information about the process is explained in Section 3.3.2. Figure 3.19 shows the region of the flame-based color ($\mathit{R}\text{eg}_{\text{color}}$) for indoor and outdoor places in the *YCbCr* color space.

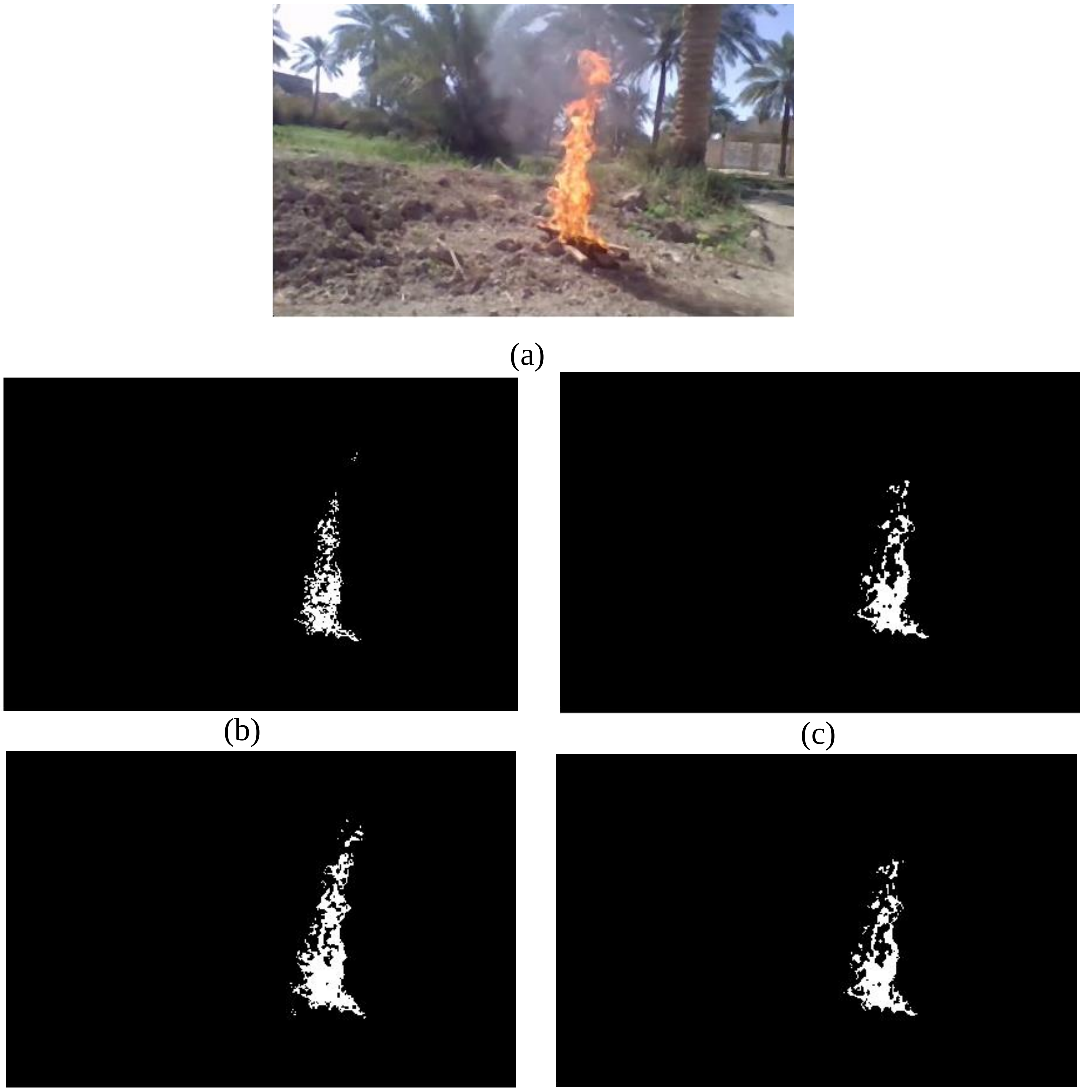


(a)

(b)

Figure 3.19 Flame detection based on YCbCr. (a.) RGB$_N$ and detected flame frames for outdoor (b) RGB$_N$ and detected flame frames for indoor.

9. Convert the frames of the flame color detection based on *HSV* and *YCbCr* color spaces to binary in order to prepare them for morphological processing and combining. As shown in Figures 3.20, 3.21, and 3.22, the resulting binary frames have white pixels representing the flame color or similar colors to the flame.



(a)

(b)

(c)

(d)

(e)

Figure 3.20 Binary frames represent the flame color in white pixels for outdoor (a.) Orginal RGB_N frame (b.) based on POS-Otsu (c.) based on POS-Kapur (d.) based on FOA-Otsu (e.) based on FOA-Kapur.

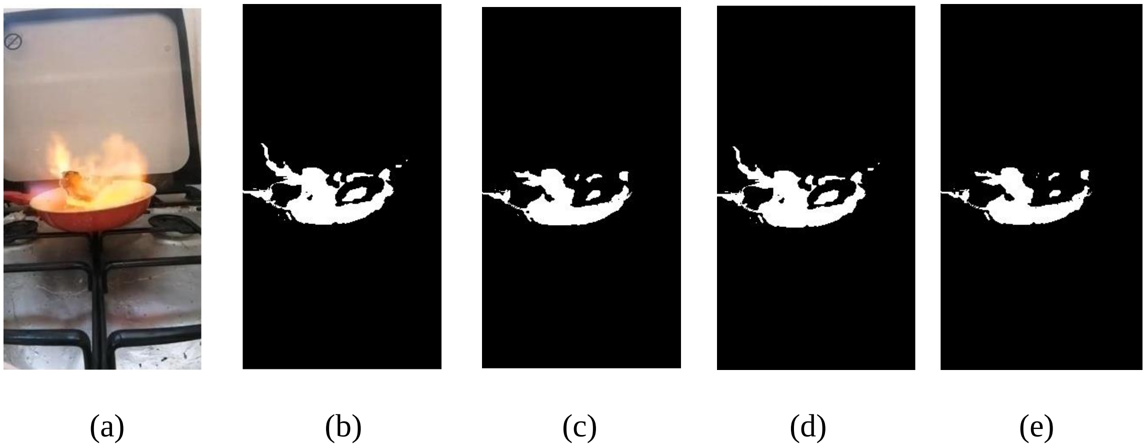


Figure 3.21 Binary frames represent the flame color in white pixels for indoor (a.) Orginal RGB_N frame (b.) based on POS-Otsu (c.) based on POS-Kapur (d.) based on FOA-Otsu (e.) based on FOA-Kapur.

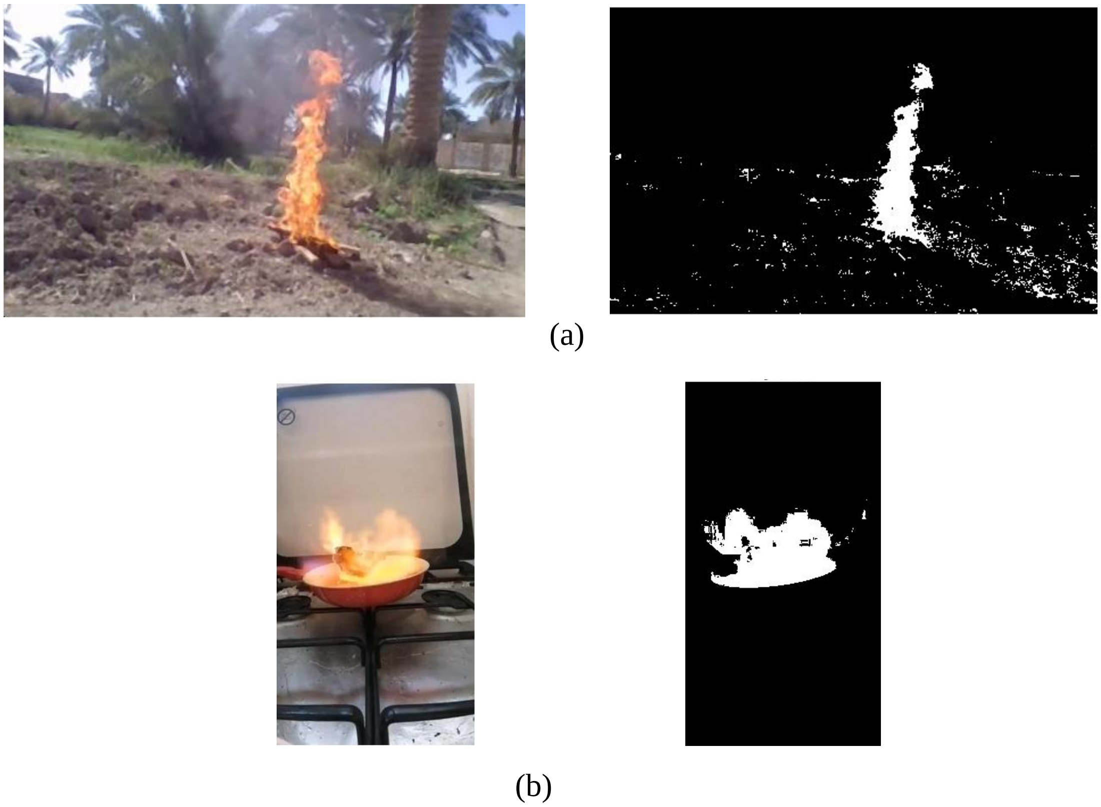


Figure 3.22 Binary frames represent the flame color in white pixels based on YCbCr for (a.) outdoor (b.) indoor.

3.4 Motion Detection

Flames, in general fires, are typically reddish in color. The color model can be made to detect flames. Regrettably, some flame-like areas in the image may have a similar color as the flame, and these flame-like regions are typically extracted from the image as real flames. Two situations cause these fire aliases: non-fire objects of a similar color as fire and backgrounds lit by fire-like light sources. In the first case, objects with a reddish color led to false extractions of flames. The second reason for incorrect fire extraction is that backgrounds illuminated by burning flames, sun reflections, and artificial light significantly impact the extraction, making the process complex and unreliable. The key to distinguishing flames and flame-colored objects from smoke and smoke-colored objects is the nature of their physical motion.

Therefore, in order to verify a real burning flame, in addition to the color function, the motion function is usually also used. These dynamic fire characteristics include sudden flame movement, changing shape, growth rate, and oscillation. Smoke and flames are turbulent phenomena. For turbulence, the chaotic nature of fire is an important feature. If the silhouettes of objects exhibit fast, time-varying behavior, this is a good sign of the presence of fire or smoke in the scene. The flame dances around the fire source, and each individual pixel in the intermittent region is only considered a fire pixel for a small fraction of the time (Yu, Mei, and Zhang 2013a). A method of frame differences is used to capture the motion of flames and smoke, which can aid in identifying real fires. As a new way, we adopt the improved frame difference method. Instead of subtracting frames in order, the subtraction is performed after eight frames, where the pixel's flame has undergone the most significant change. To detect the motion of the flame and smoke in a proposed system, these steps have been performed as shown in Figure 3.23:

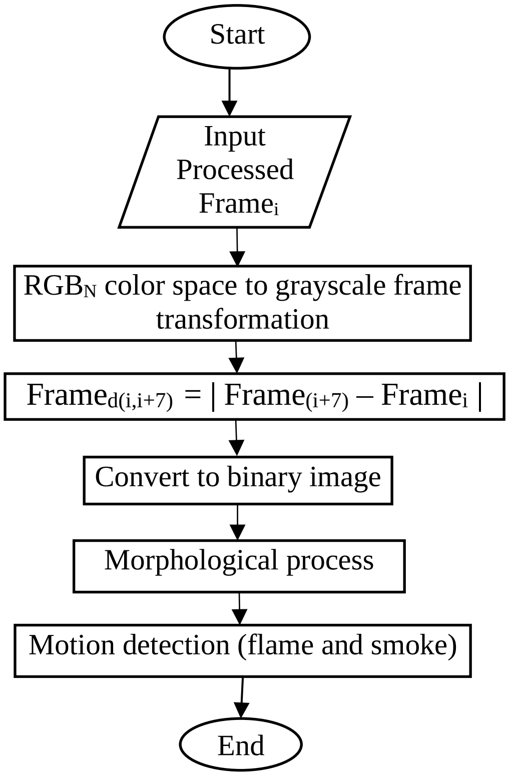
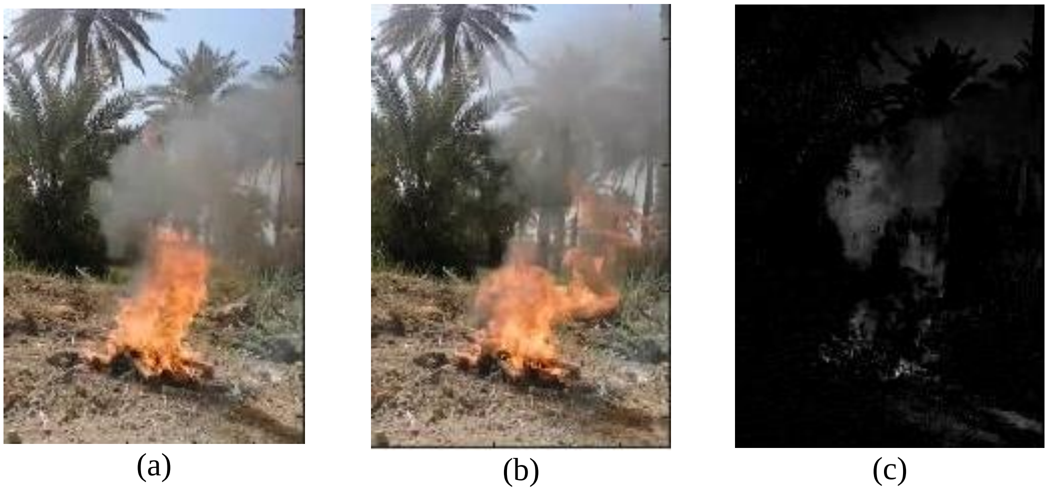

Start
Input Processed Frame$_i$
RGB$_N$ color space to grayscale frame transformation
Frame$_{d(i,i+7)}$ = | Frame$_{(i+7)}$ – Frame$_i$ |
Convert to binary image
Morphological process
Motion detection (flame and smoke)
End

Figure 3.23 Flowchart of the fire motion detection

1. The color space transformation from RGB_N color space to gray frame, for which the transformation equation is written in Equations (2.2).
2. The subtraction is performed after eight frames, during which the pixel's color changes significantly. The value selected based on the trial-and-error method.

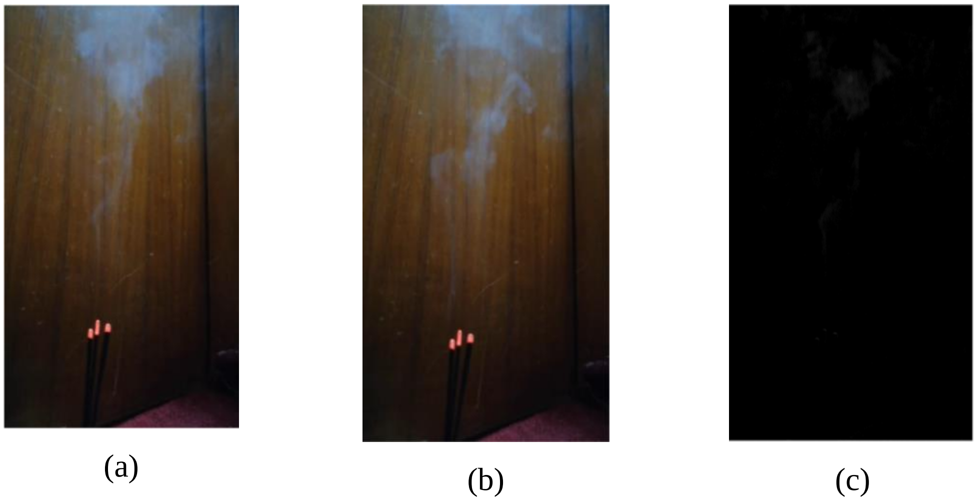
$$Frame_{d(i,i+7)} = |Frame_{(i+7)} - Frame_i| \qquad \textbf{(3.19)}$$

In the video, the value of the i^{th} the frame is represented as $Frame_i$. Similarly, the value of the $(i+7)^{th}$ frame is represented as $Frame_{(i+7)}$. The differences between the two frames in Figures 3.24 - 3.27 indicate smoke and flame movement in indoor and outdoor places.



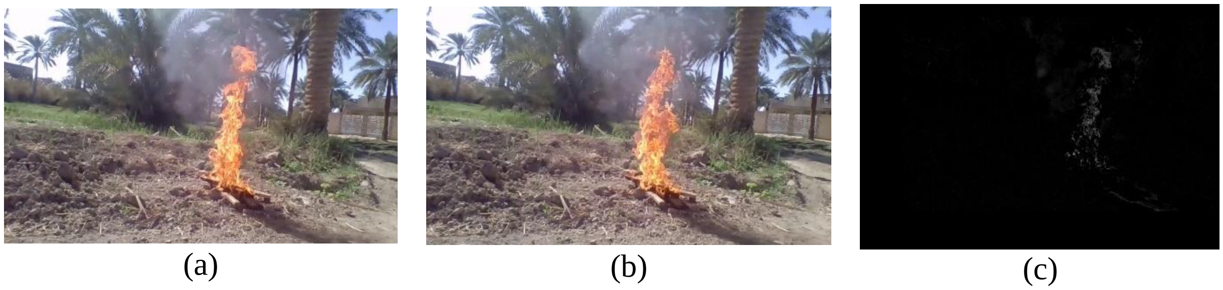
(a) (b) (c)

Figure 3.24 The frame differences result for outdoor. (a.) RGB$_N$ Frame$_1$. (b.) RGB$_N$ Frame 8. (c.) the difference between a and b.



(a) (b) (c)

Figure 3.25 The frame differences result for indoor. (a.) RGB$_N$ Frame$_1$. (b.) RGB$_N$ Frame 8. (c.) the difference between a and b.



(a) (b) (c)

Figure 3.26 The frame differences result for outdoor. (a.) RGB$_N$ Frame$_1$. (b.) RGB$_N$ Frame 8. (c.) the difference between a and b.

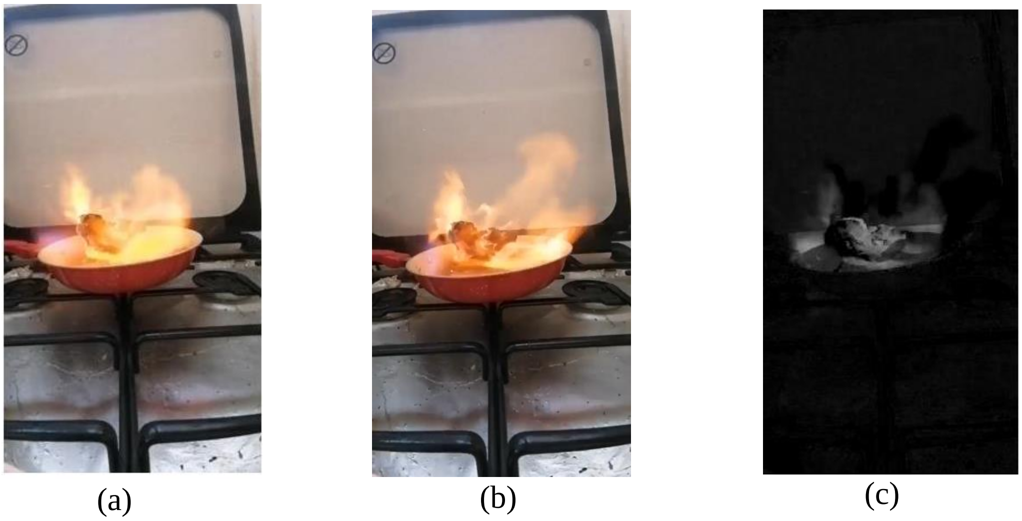
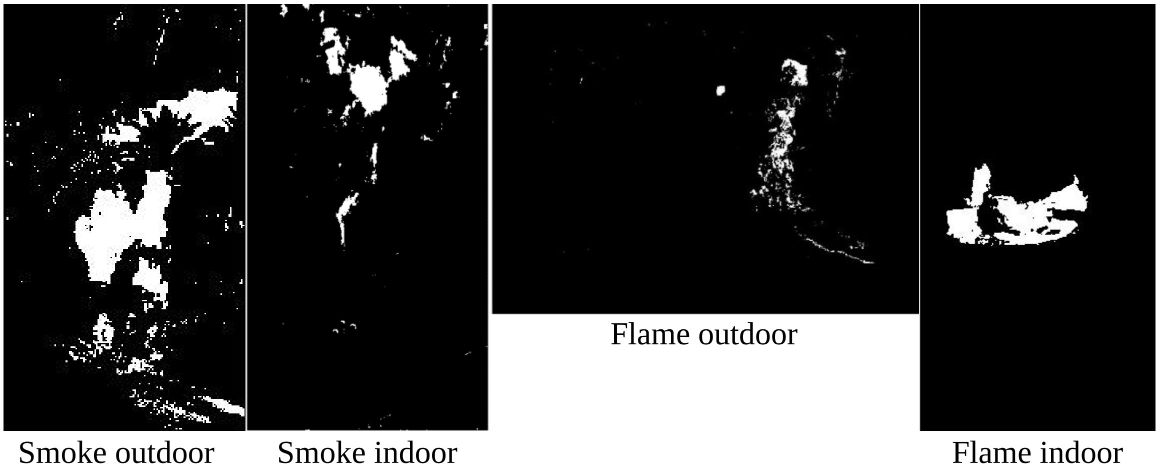


Figure 3.27 The frame differences result from indoor. (a.) RGB_N Frame$_1$. (b.) RGB_N Frame 8. (c.) the difference between a and b.

3. Convert the frame to binary to qualify for the morphological processes and combine it with color detection. In Figure 3.28, the binary frames depict white pixels indicating smoke and flame movement in indoor and outdoor places.


Smoke outdoor

Smoke indoor

Flame outdoor

Flame indoor

Figure 3.28 A binary frame represents the motion in for smoke and flame movement in indoor and outdoor places from Figures 3.24, 3.25, 3.26, and 3.27, respectively.

3.5 Morphological Process

The post-processing step involves closing, opening, and filling techniques. The morphological operations aim to fill gaps and remove small and unwanted objects. Figures 3.29 – 3.34 illustrate the morphological processing applied to color and motion detection for flame and smoke in outdoor and indoor places.

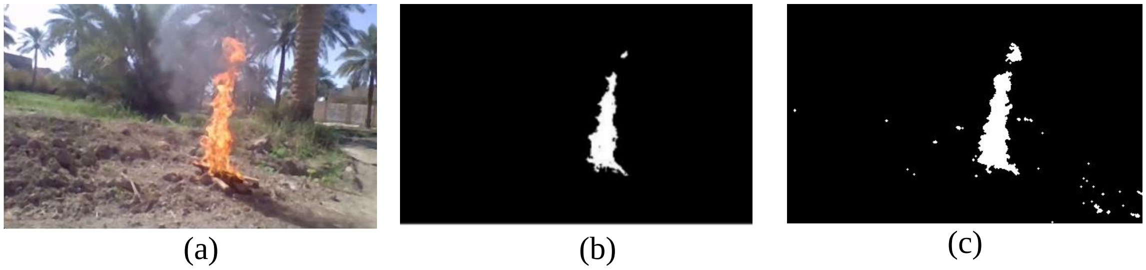

(a) (b) (c)

Figure 3.29 The morphological process for flame color detection by non-adaptive thresholds in outdoor places. (a.) Orginal RGB$_N$ frame (b.) based on HSV (c.) based on YCbCr

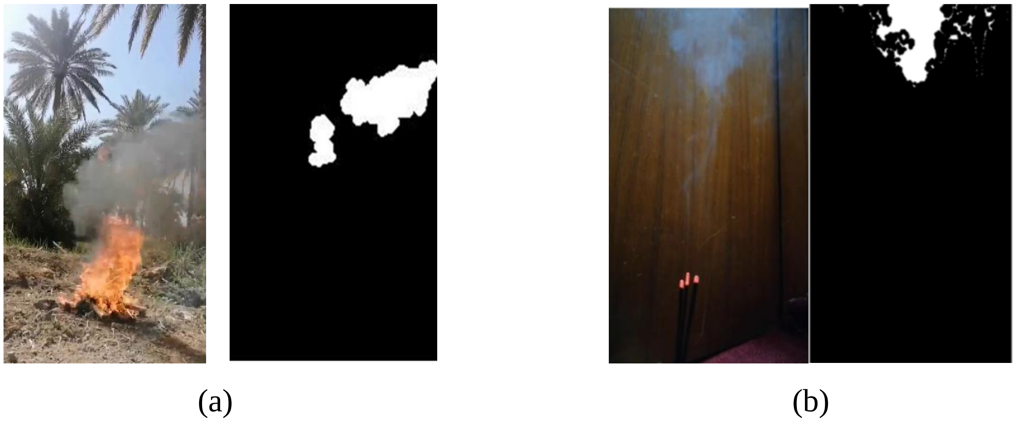
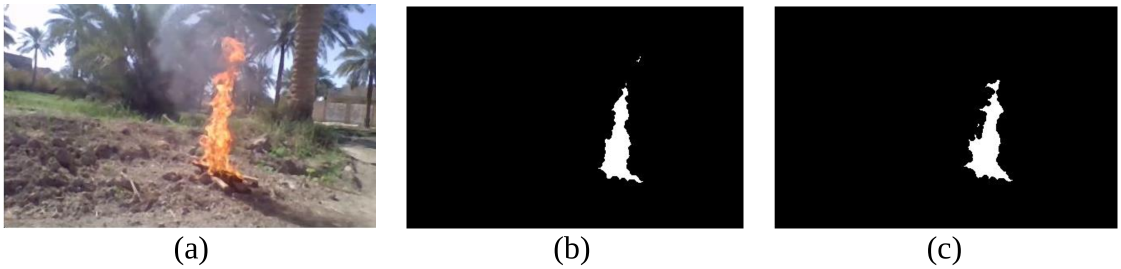
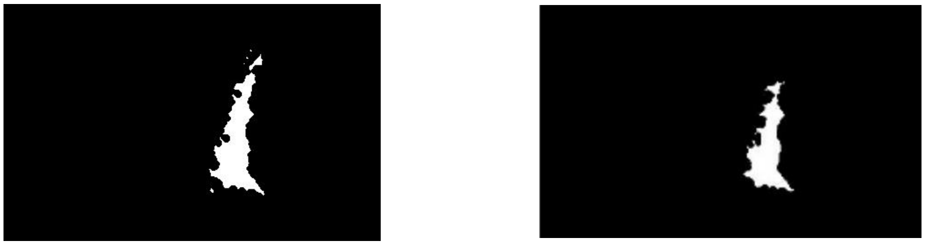

(a) (b)

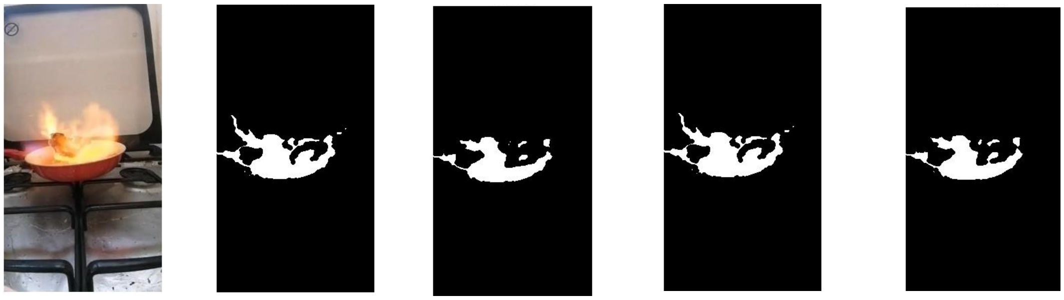
Figure 3.30 The morphological process for smoke color detection-based CIE L*a*b by static thresholds for (a.) outdoor place (b.) indoor place.


(a) (b) (c)



(d) (e)

Figure 3.31 The morphological process of the adaptive flame detection for outdoor (a.) Orginal RGB_N frame (b.) based on POS-Otsu (c.) based on POS-Kapur (d.) based on FOA-Otsu (e.) based on FOA-Kapur.



(a) (b) (c) (d) (e)

Figure 3.32 The morphological process of the adaptive flame detection for indoor (a.) Orginal RGB_N frame (b.) based on POS-Otsu (c.) based on POS-Kapur (d.) based on FOA-Otsu (e.) based on FOA-Kapur.

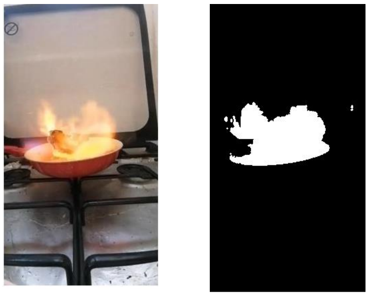


Figure 3.33 The morphological process of the proposed flame detection for indoor based on YCbCr.

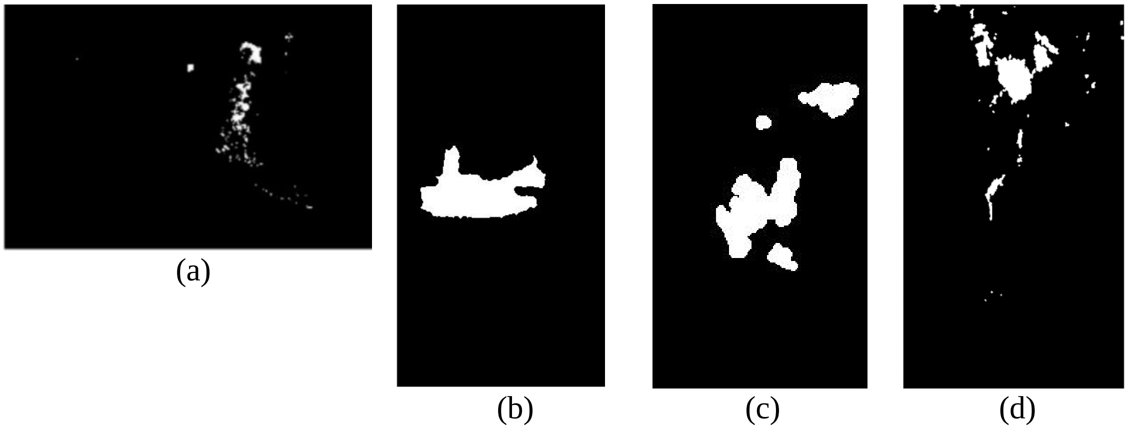


Figure 3.34 The morphological process for motion detection. (a.) flame outdoor (b.) flame indoor (c.) smoke outdoor (d.) smoke indoor

The next step involves combining the two-color detection methods for flames to identify the regions of interest where the fire is present, referred to as Reg_{color} *(J, K, i)*. The *HSV* color space detection at the static threshold is combined with the *YCbCr* color space by bitwise *AND* in the first method for flame color detection. The second method combines the *HSV* color space using various techniques, including *POS* and *FOA*, with the *YCbCr* color space using a bitwise *AND* operation to produce four results. Figures 3.35 - 3.43 show the combination of two-color flame detection for indoor and outdoor places.

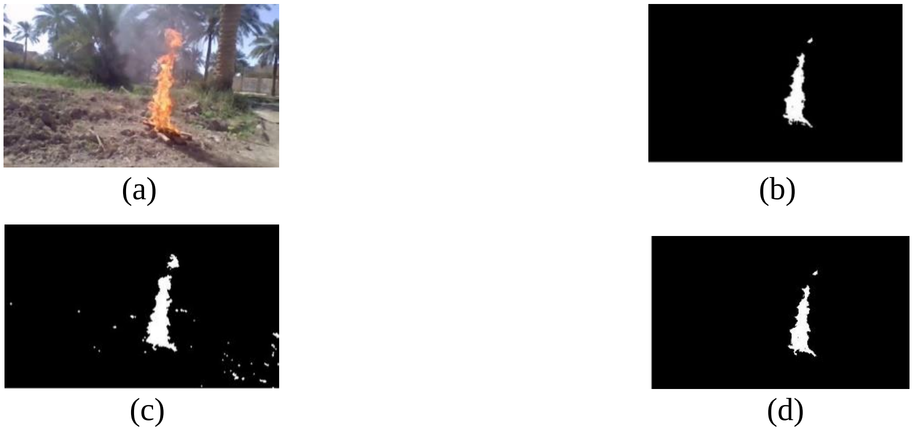


Figure 3.35 The combination of two colors. (a.) Outdoor place in RGB_N - $Frames_1$. (b. and c.) The color of flame detection based HSV and YCbCr by non-adaptive thresholds. (d.) The combination between b and c.

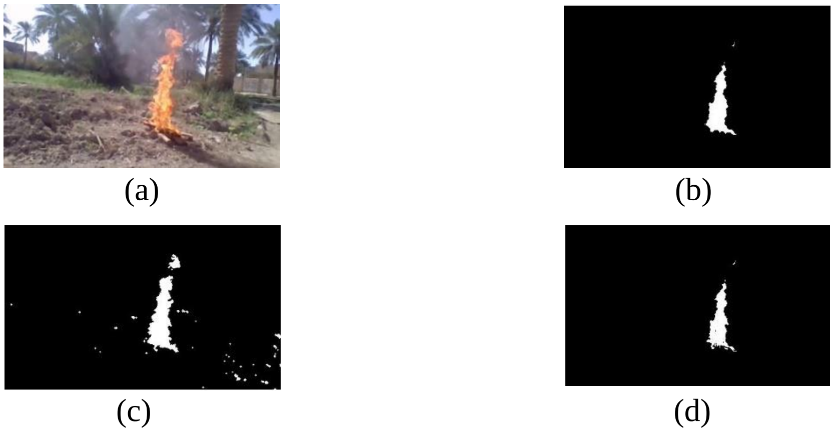


Figure 3.36 The combination of two colors of flame. (a.) Outdoor place in RGB$_N$ -Frames$_1$. (b. and c.) based HSV for POS-Otsu and YCbCr (d.) The combination between b and c.

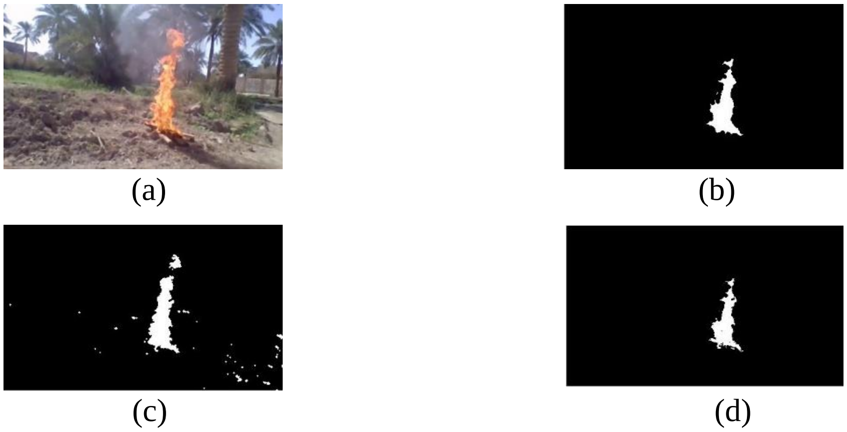


Figure 3.37 The combination of two colors of flame. (a.) Outdoor place in RGB$_N$ -Frames1. (b. and c.) based HSV for POS-Kapur and YCbCr (d.) The combination between b and c.

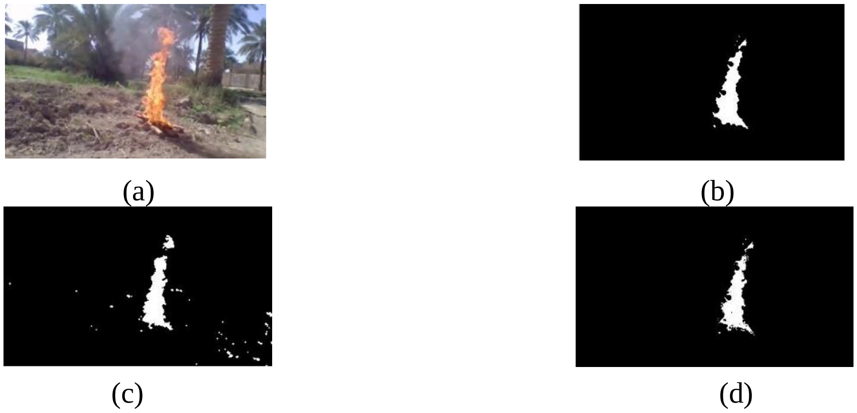
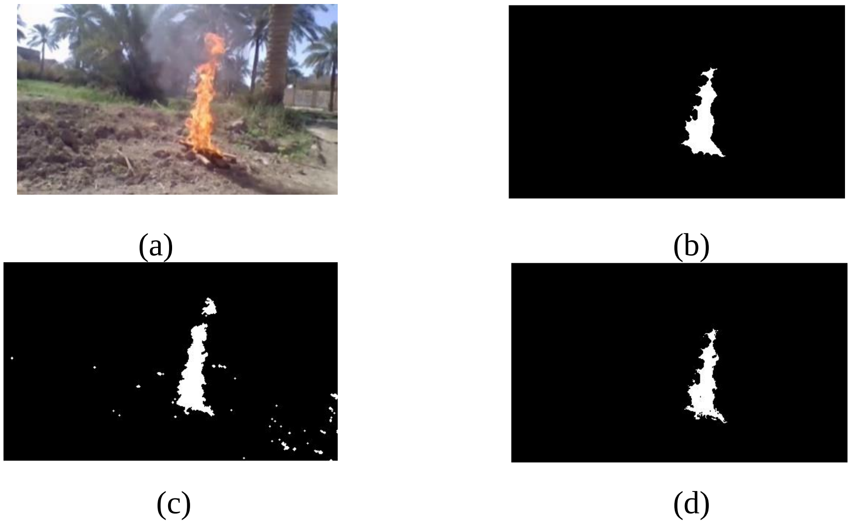


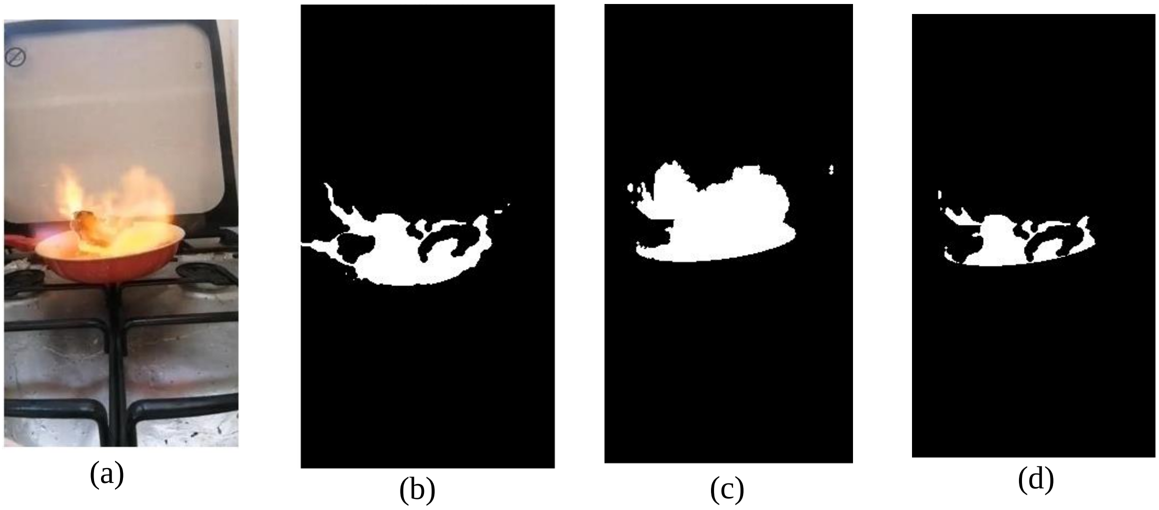
Figure 3.38 The combination of two colors of flame. (a.) Outdoor place in RGB$_N$ Frames$_1$. (b. and c.) based HSV for FOA-Otsu and YCbCr (d.) The combination between b and c.



(a) (b)

(c) (d)

Figure 3.39 The combination of two colors of flame. (a.) Outdoor place in RGB_N -Frames_1. (b. and c.) based HSV for FOA-Kapur and YCbCr (d.) The combination between b and c.



(a) (b) (c) (d)

Figure 3.40 The combination of two colors of flame. (a.) Indoor place in RGB_N -Frames_1. (b. and c.) based HSV for POS-Otsu and YCbCr colors (d.) The combination between b and c.

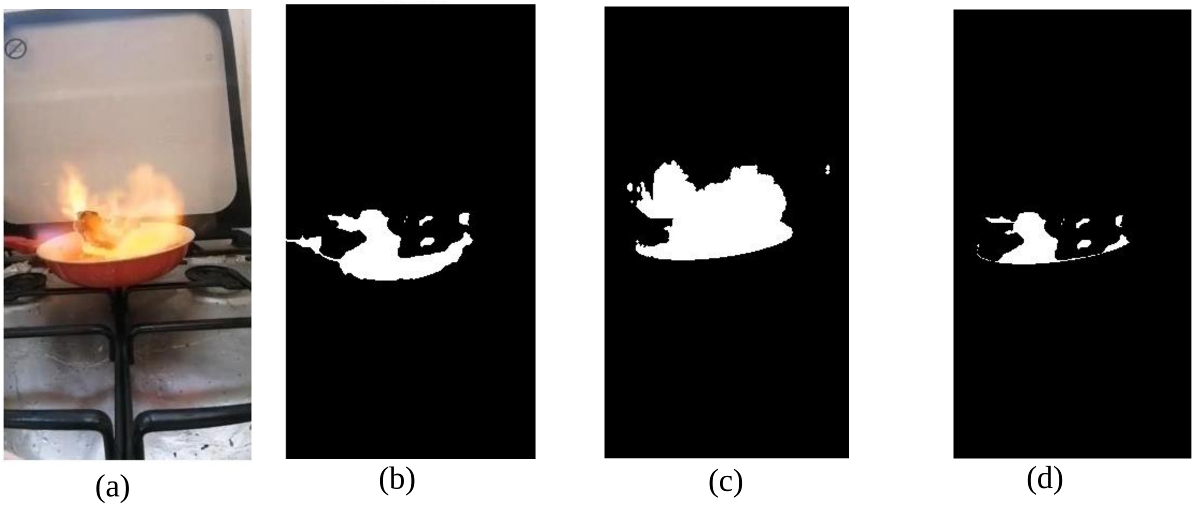


Figure 3.41 The combination of two colors of flame. (a.) Indoor place in RGB$_N$ -Frames$_1$. (b. and c.) detection-based HSV for POS-Kapur and YCbCr colors space by non-adaptive thresholds. (d.) The combination between b and c.

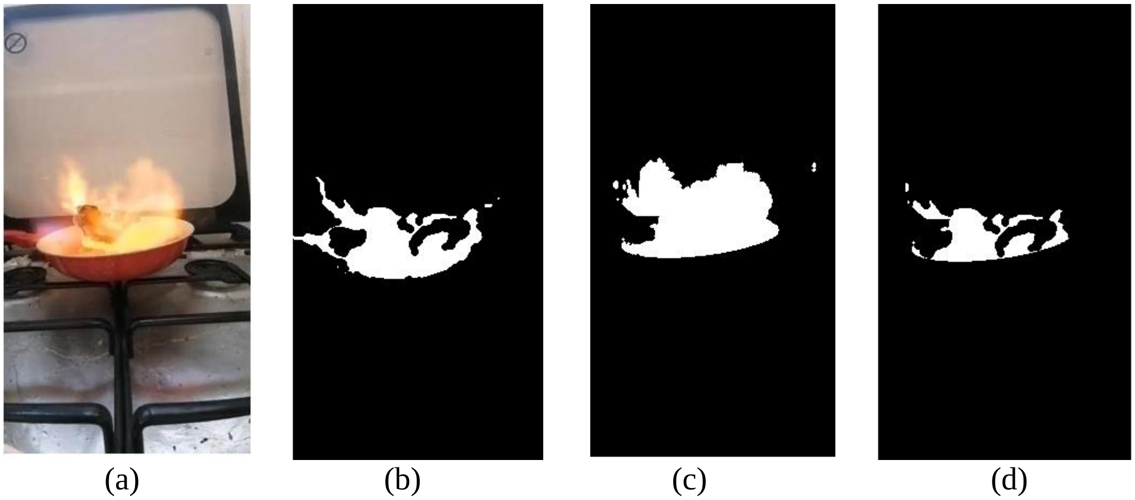


Figure 3.42 The combination of two colors of flame. (a.) Indoor place in RGB$_N$ -Frames$_1$. (b. and c.) based HSV for FOA-Otsu and YCbCr (d.) The combination between b and c.

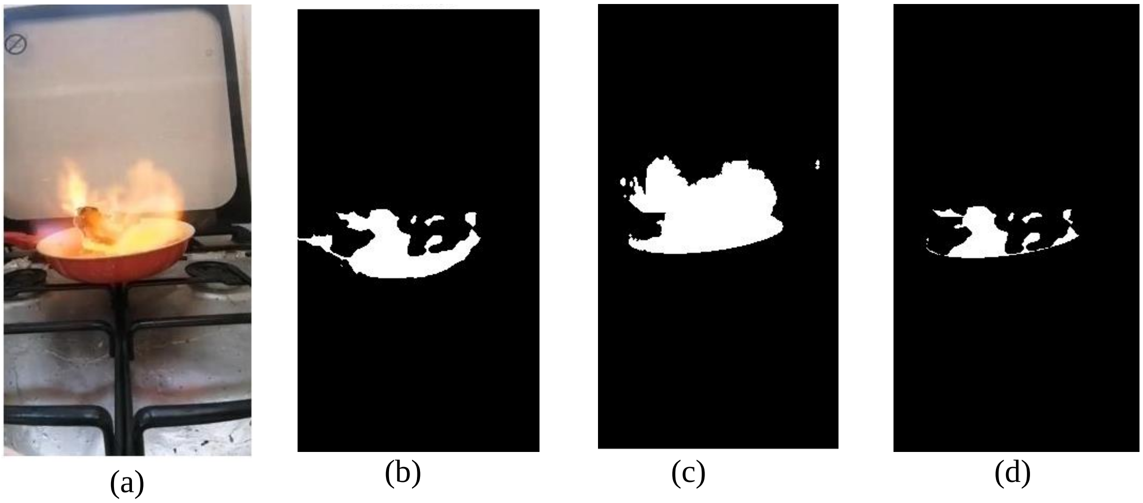


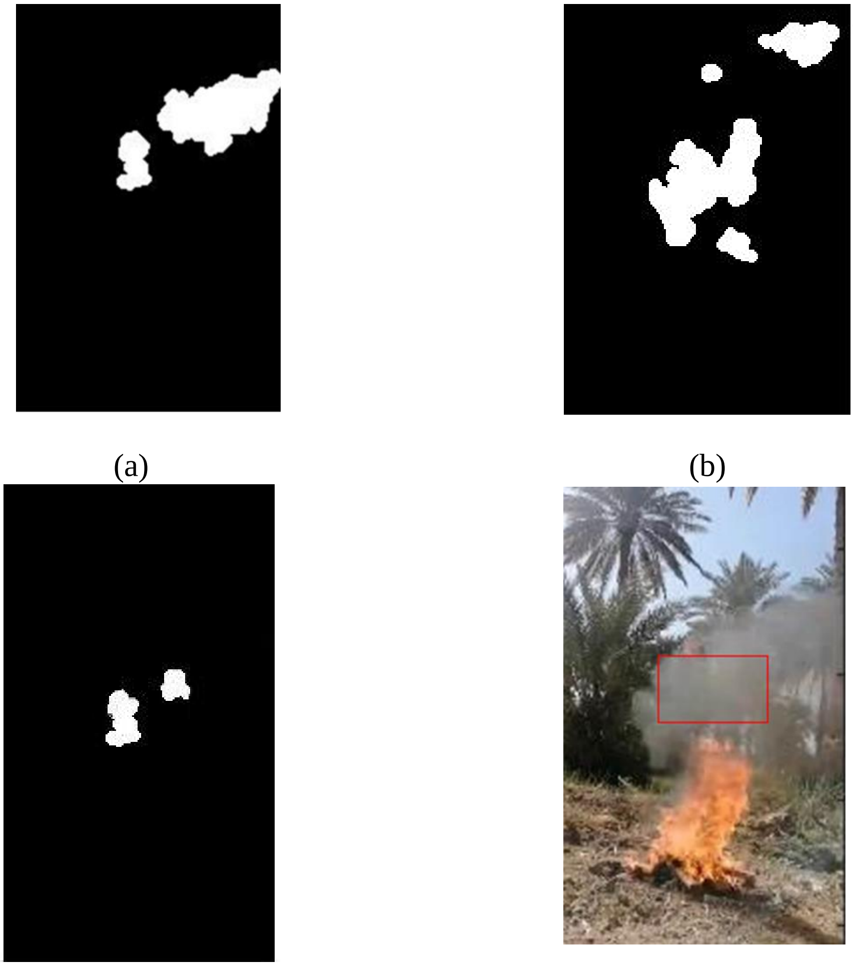
Figure 3.43 The combination of two colors of flame. (a.) Indoor place in RGB$_N$ -Frames$_1$. (b. and c.) based HSV for FOA-Kapur and YCbCr (d.) The combination between b and c.

3.6 Fire Area Detection

After thoroughly examining the frame difference and color feature of fire, it has been determined that using frame difference or color detection alone to identify fire would result in a high rate of false alarms. Therefore, additional operations that combine the results of both methods must be performed effectively to utilize their properties and accurately identify the fire region. In additional, the result of the combination is no longer sequential due to the improved frame difference method. This method allows fire detection in just four frames within one second instead of the usual 30 frames, so the method is called selected frame N_s. The new technique speeds up the processing time, maintains accuracy, and provides timely alarm notifications. Figures 3.44 - 3.54 show the combination between the color and motion fire regions Reg_{fire} for all methods by Bitwise AND.

$$Reg_{fire}(J,K,i) = Reg_{color}(J,K,i) \cap Frame_d(J,K,i) \qquad \textbf{(3.20)}$$

When the fire region has been determined, the fire is bound in binary by specifying the regions of the flame or smoke and then inserting the green bounding box for the flame and the red box for the smoke, as shown in Figures 3.45 - 3.55. Finally, the area of the bounded region is computed, which is calculated by subtracting the original frame from the bounding frame as a new and simple method. When the results of this area are greater than the threshold value, they are considered a fire. The area's threshold is not less than 55 for flame and 85 for smoke. These values selected based on the trial-and-error method.



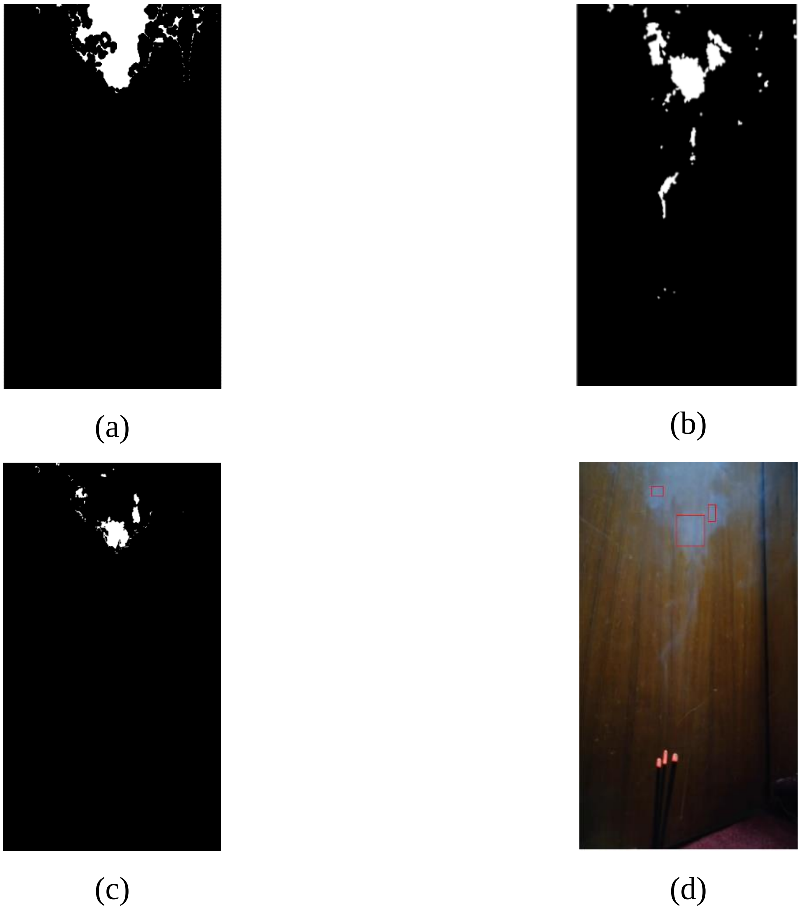
(a)

(b)

(c)

(d)

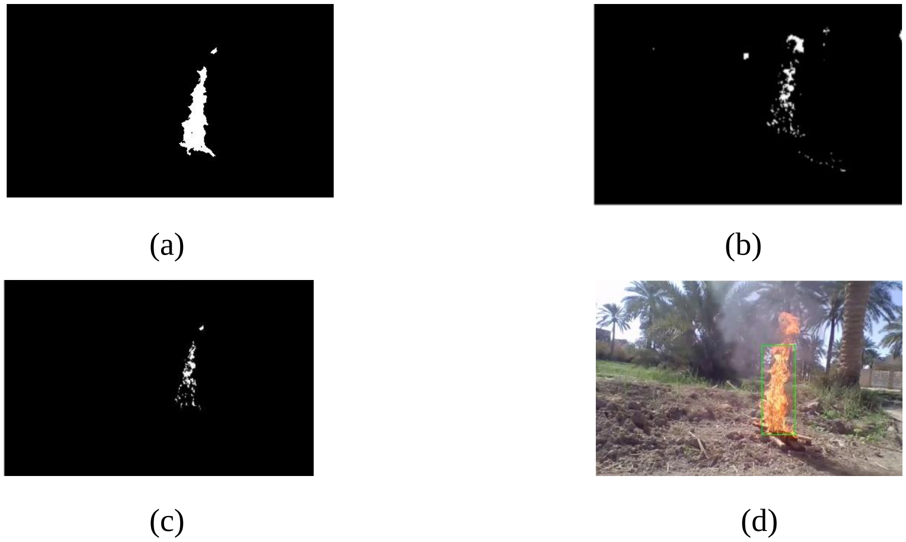
Figure 3.44 Detection of smoke outdoors (a.) colors of smoke detection (b.) motion of smoke (c.) combination between a and b. (d.) result in the red box



(a) (b)

(c) (d)

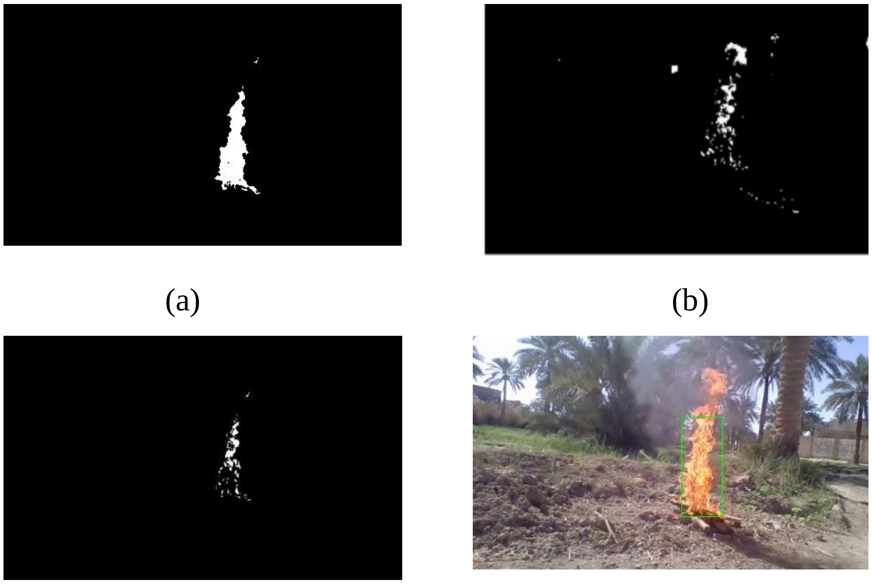
Figure 3.45 Detection of smoke indoors (a.) colors of smoke detection (b.) motion of smoke (c.) combination between a and b. (d.) result in the red box



(a) (b)

(c) (d)

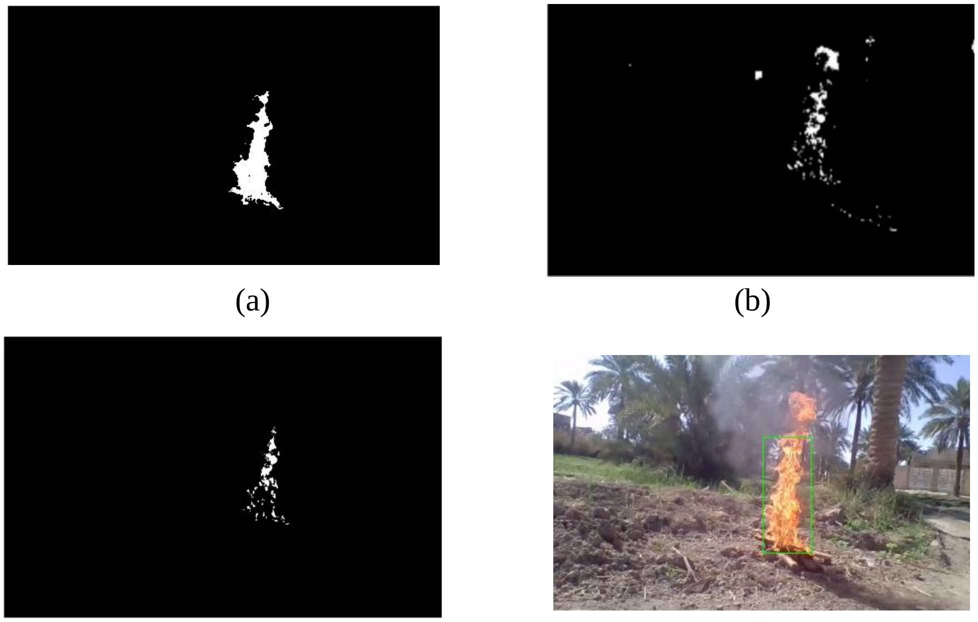
Figure 3.46 Detection of non-adaptive flame outdoors by (a.) colors of flame detection HSV\YCbCr (b.) motion of flame (c.) combination between a and b. (d.) result in a green box



(a) (b)

(c) (d)

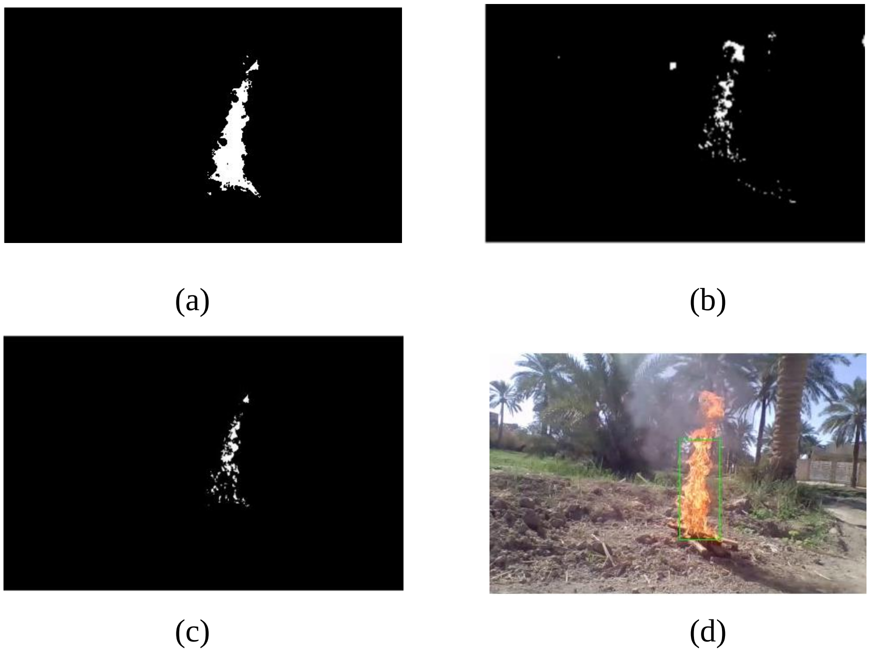
Figure 3.47 Detection of adaptive flame outdoors (a.) colors of flame detection POS-Otsu\YCbCr (b.) motion of flame (c.) combination between a and b. (d.) result in a green box



(a) (b)

(c) (d)

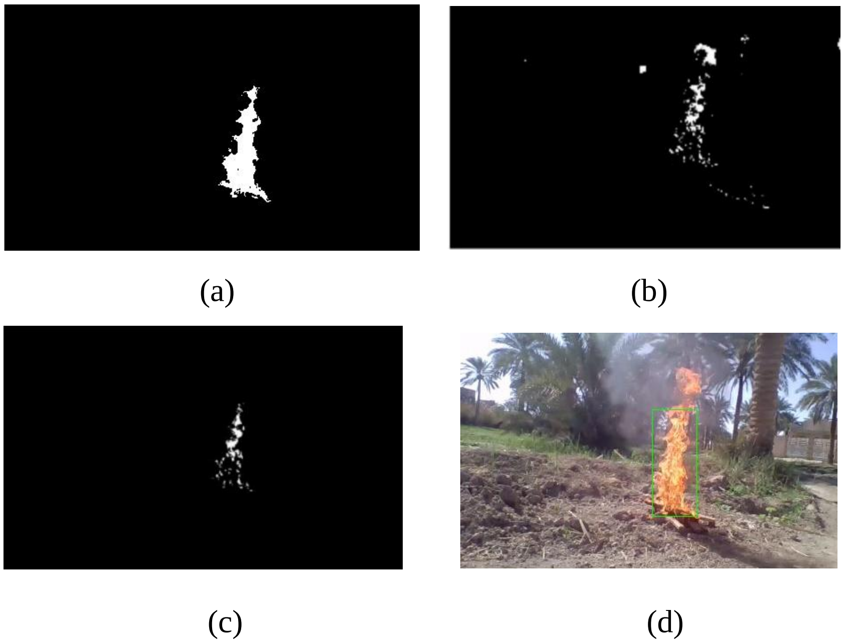
Figure 3.48 Detection of adaptive flame outdoors (a.) colors of flame detection POS-Kapur\YCbCr (b.) motion of flame (c.) combination between a and b. (d.) result in a green box



(a) (b)

(c) (d)

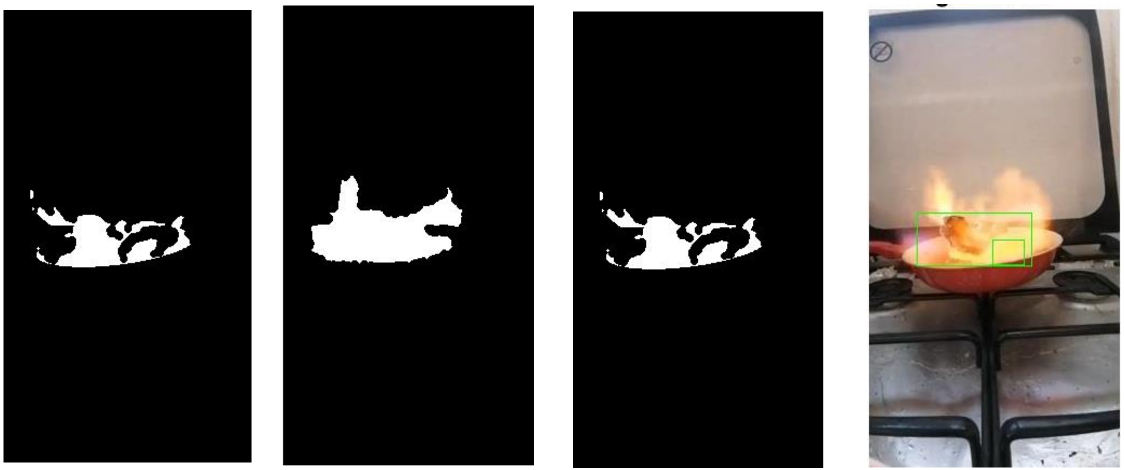
Figure 3.49 Detection of adaptive flame outdoors (a.) colors of flame detection FOA-Otsu\YCbCr (b.) motion of flame (c.) combination between a and b. (d.) result in a green box.



(a) (b)

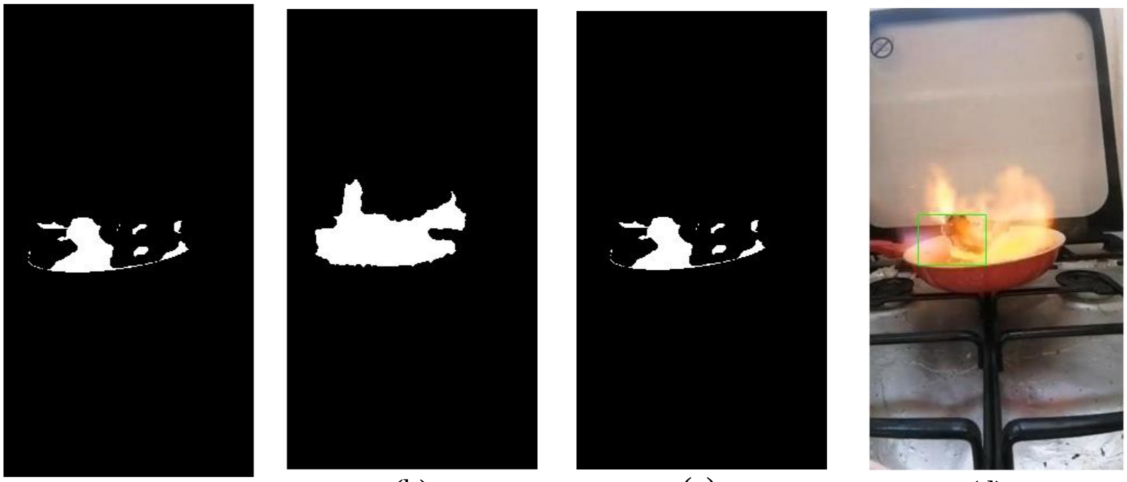
(c) (d)

Figure 3.50 Detection of adaptive flame outdoors (a.) colors of flame detection FOA-Kapur\YCbCr (b.) motion of flame (c.) combination between a and b. (d.) result in a green box



(a) (b) (c) (d)

Figure 3.51 Detection of adaptive flame indoors (a.) colors of flame detection POS-Otsu\YCbCr (b.) motion of flame (c.) combination between a and b. (d.) result in a green box.



(a) (b) (c) (d)

Figure 3.52 Detection of adaptive flame indoors (a.) colors of flame detection POS-Kapur\YCbCr (b.) motion of flame (c.) combination between a and b. (d.) result in a green box.

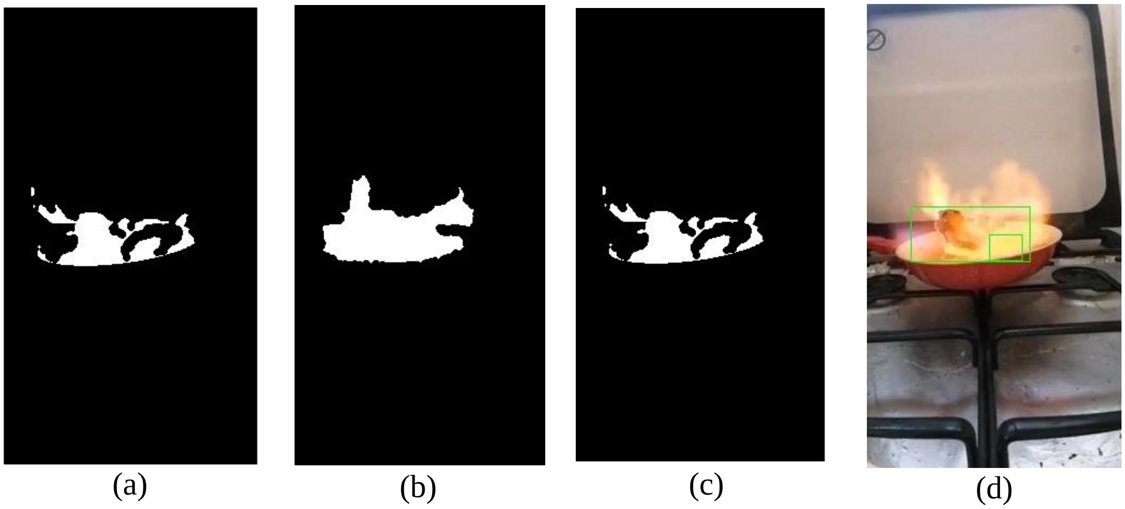


Figure 3.53 Detection of adaptive flame indoors by (a.) colors of flame detection FOA-Otsu\YCbCr (b.) motion of flame (c.) combination between a and b. (d.) result in a green box.

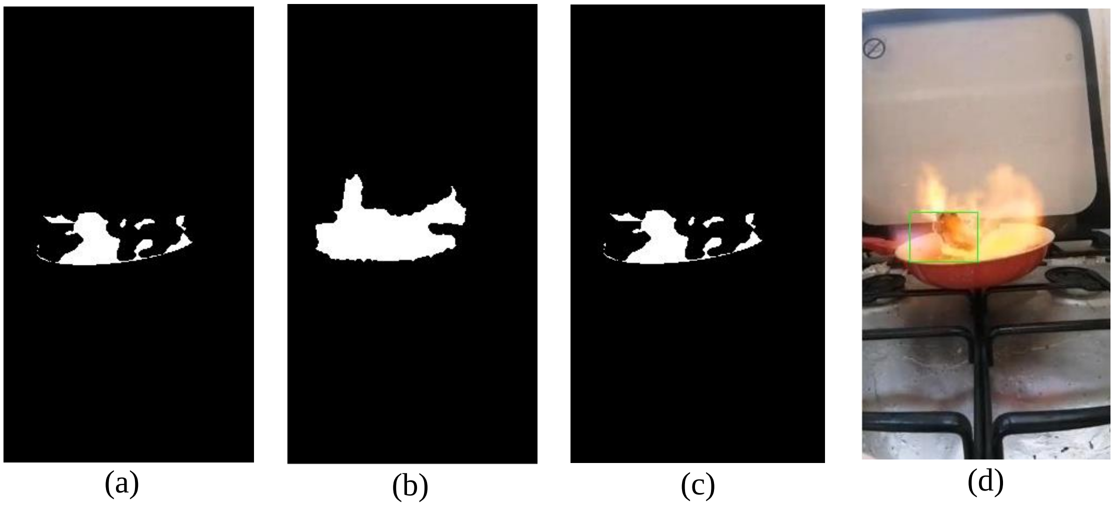


Figure 3.54 Detection of adaptive flame indoors (a.) colors of flame detection FOA-Kapur\YCbCr (b.) motion of flame (c.) combination between a and b. (d.) result in a green box

3.7 Fire Localization

Fire localization refers to identifying a fire's location within a building or outdoor place. This is a critical task for firefighters and other emergency responders, as accurate information about the location of a fire can help them more quickly and effectively deploy resources to contain and extinguish the blaze.

In real-time, the flame and smoke detection systems provide the fire's position in the current frame in pixels. So, to determine the site of the fire in the actual world, a real-world fire localization method comprising two steps is introduced.

Our two-stage method for identifying the location of a fire in real-time is depicted in Figure 3.55. The calibration of the camera is the primary goal of the first step of the fire-localization. Camera calibration is the process of determining a camera's intrinsic and extrinsic parameters. Intrinsic parameters refer to camera-specific properties, such as principal point, focal length, and distortion coefficients, denoted in pixels by the *fx* and *fy* notation for the focal length and *cx* and *cy* for the principal points. Extrinsic parameters describe the camera's position and orientation in the 3D world, which involves coordinating system conversions from 3D world coordinates toward 3D camera coordinates. The rotation matrix *Rot* is used to perform a rotation in Euclidean space. The transformation matrix *Tr* represents the camera-centered coordinate system's translation of the world coordinate system's origin. The transformation matrix *Tr* is typically a 3 × 1 matrix.

In comparison, the rotation matrix is a 3 × 3 matrix. Assume *J* and *K* represent the position of the fire in the video in pixels, and *Xrw*, *Yrw*, and *Zrw* represent where the fire is located in the real world. Equation (3.21) is satisfied by the relationship between (*Xrw*, *Yrw*, *Zrw*) and (*J*, *K*) by the projective transformation (Z. Zhang 2021).

$$\begin{bmatrix} J \\ K \\ 1 \end{bmatrix} = \begin{bmatrix} fx & 0 & cx \\ 0 & fy & cy \\ 0 & 0 & 1 \end{bmatrix} \left(Rot \begin{bmatrix} Xrw \\ Yrw \\ Zrw \end{bmatrix} + Tr \right) \quad \textbf{(3.21)}$$

Furthermore, actual lenses usually have distortion. After the camera calibration, radial distortion has been removed. The radical distortion coefficient can make the localization of fire more accurate.

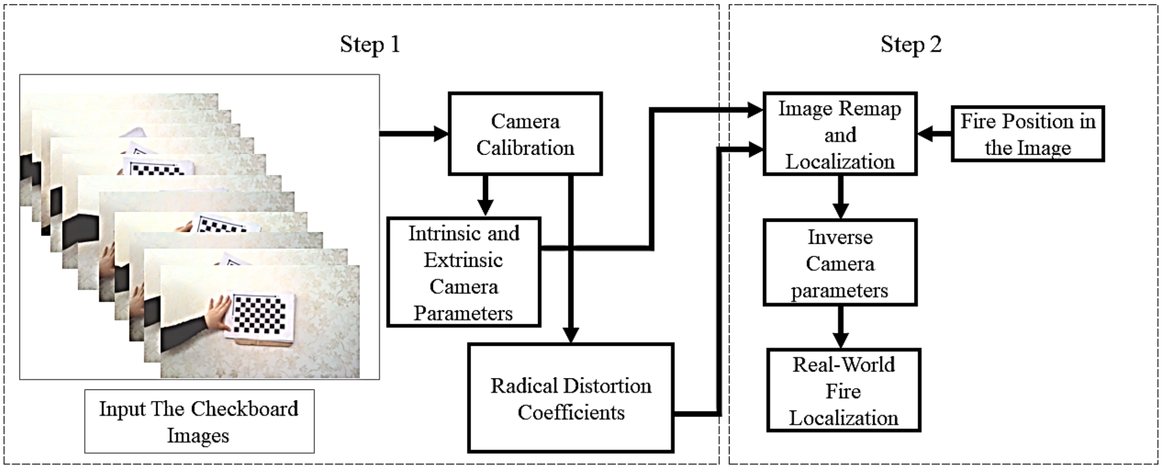

Step 1
Step 2
Camera Calibration
Intrinsic and Extrinsic Camera Parameters
Radical Distortion Coefficients
Input The Checkboard Images
Image Remap and Localization
Fire Position in the Image
Inverse Camera parameters
Real-World Fire Localization

Figure 3.55 The proposed fire localization method.

The following are the steps involved in the fire localization system:

1. Use a calibration pattern (checkerboard technique) to calibrate the camera. This pattern should be placed at different distances from the camera to calculate the camera's intrinsic and extrinsic parameters accurately.
2. Capture multiple images of more than 10 images of the calibration pattern from different angles and positions. Make sure that the pattern is fully visible in all images.
3. Use camera calibration software MATLAB (Using the Single Camera Calibrator App - MATLAB & Simulink 2023) to calibrate the camera using the images captured in step 2. This will give the intrinsic and extrinsic parameters of the camera and radial distortion.

4. From the previous flame and smoke detection systems, the fire position in each frame is identified according to the centroid of the bounding box.
5. The fire position point in pixels is mapped, and localization.
6. Find the inverse of the intrinsic and rotation matrix (Z. Zhang 2021):

$$\begin{bmatrix} Xrw \\ Yrw \\ Zrw \end{bmatrix} = \begin{bmatrix} fx & 0 & cx \\ 0 & fy & cy \\ 0 & 0 & 1 \end{bmatrix}^{-1} Rot^{-1}(\begin{bmatrix} J \\ K \\ 1 \end{bmatrix} - Tr) \qquad \textbf{(3.22)}$$

The real-world point of fire localization is computed from Equation (3.22); locating this point depends on the accuracy of camera parameters.

Chapter Four: Results and Discussion

4.1 Introduction

Implementing of a fire detection system is crucial to detecting and mitigating the potential harm that fires can cause. These systems can greatly enhance the safety of people, property, and the environment. In this chapter, the experimental results of the work are presented, analyzed, and compared to other related work. The proposed system has been evaluated for its performance, including fire detection accuracy and reliability of the systems. Furthermore, the four elements of the confusion matrix (*TPs*, *TNs*, *FPs*, and *FNs*) are calculated, as shown in Section 2.11, for comparison with the other related work.

The system has been tested using three groups of datasets from trusted sites as well as our own dataset. Both offline and online experimental tests have been achieved to examine the suitability of the proposed system for real-time utilization.

4.2 Materials and datasets

MATLAB R2021b, operating on a Windows 10 system for PC laptops, is used to implement the proposed fire detection system. The laptop has an Intel (R) Core (TM) i7 2.70GHz processor and 16GB of RAM.

A laptop camera has a resolution of 1280 × 720 and a frame rate of 30 frames per second is utilized for real-time fire detection. Two main groups of video datasets are used in this work, from internet sites and our own. The downloaded datasets include 92 fire videos, either smoke based, flame-based, or both (A.E. Çetin 2014; Grammalidis, Dimitropoulos, and Cetin 2017;

KMU Fire & Smoke Database 2012). The own recorded datasets are either captured by a smartphone or Laptop camera.

These tested videos consist of both actual fire and objects resembling fire, such as a sunny field and people wearing clothes colors similar to flame and smoke. Also, the tested videos have indoor and outdoor environments.

4.3 Results of Smoke Detection

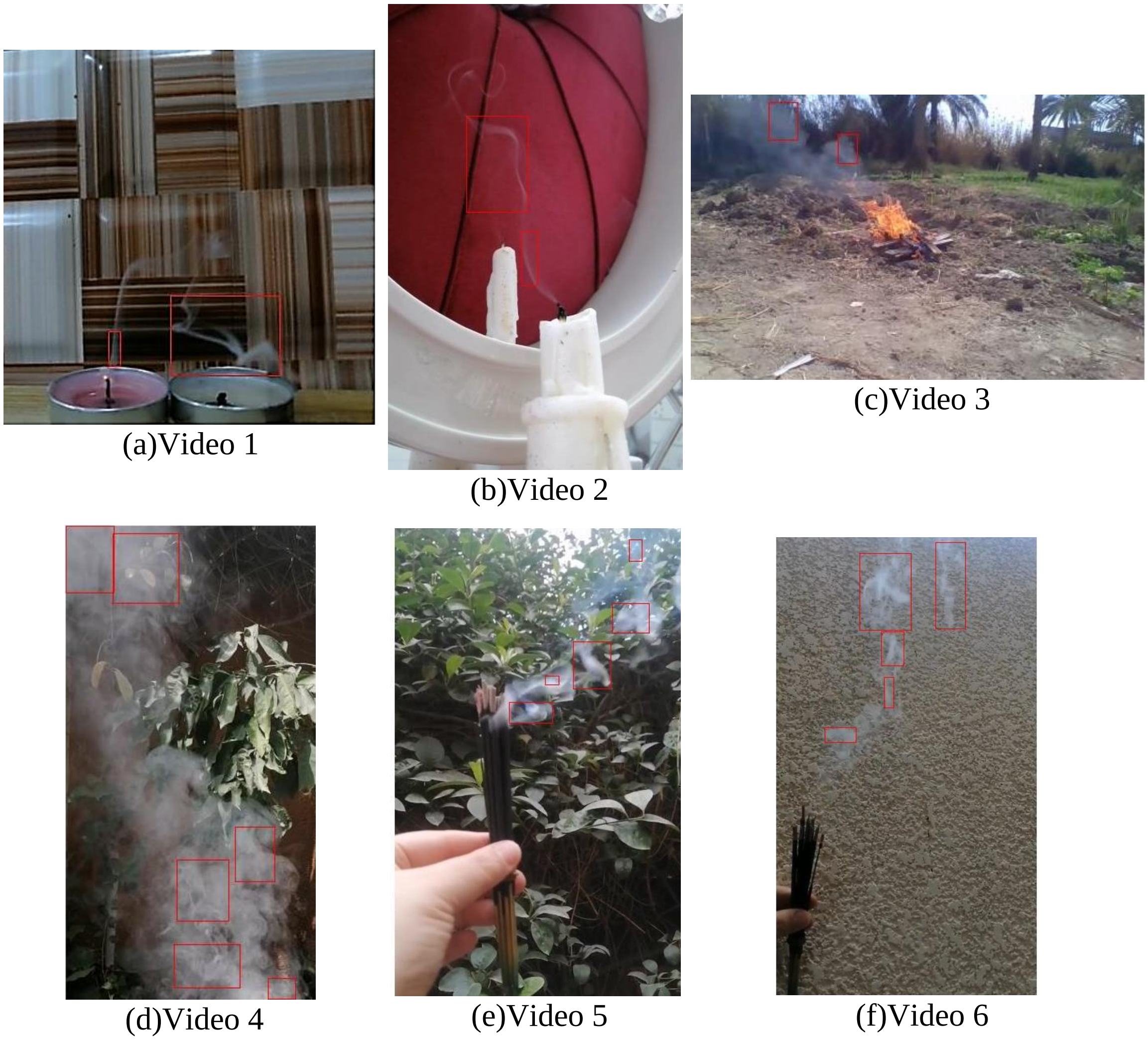
Smoke detection results for indoor and outdoor environments are categorized into real-time and recorded video results.

4.3.1 Real-time Smoke Detection Results

Real-time tests of smoke detection for various indoor and outdoor places are presented in Figure 4.1. Six different smoke videos in different regions are used in tests. The videos show smoke in various colors, ranging from gray to white, as bound in the red boxes for each video shown in Figure 4.1. Table 4.1 summarizes the real-time experiment results. The average accuracy of the smoke detection system for all six videos computed according to Equation (2.37) is approximately 93%. The average system accuracy for indoors is 90%, while outdoor accuracy is 94.1%. The contrast between indoor and outdoor accuracy is due to the effect of changing the light on the smoke.

The proposed system on Figure 4.1 (*Video 1)* and (*Video 2)* demonstrate the system's ability to detect the very low density of the smoke. Also, the system can detect different colors of smoke, such as the test video (*Video 3*) and (*Video 5*). Furthermore, the system can distinguish smoke from smoke-colored backgrounds, as seen in Figure 4.1 *(Video 6),* preventing false detections.

The timely identification of fires is of utmost importance, and the system being proposed effectively addresses this challenge by swiftly detecting smoke in real-time. By integrating the *LWT* into the system, the response time has been significantly improved to less than 0.08 seconds. This enhancement has led to a substantial reduction in the overall time required for smoke detection, effectively reducing it by more than half from the previous duration of 1.6 seconds.



(a)Video 1

(b)Video 2

(c)Video 3

(d)Video 4

(e)Video 5

(f)Video 6

Figure 4.1 Test results of the real-time smoke detection (a and b) indoors. (c-f) outdoors.

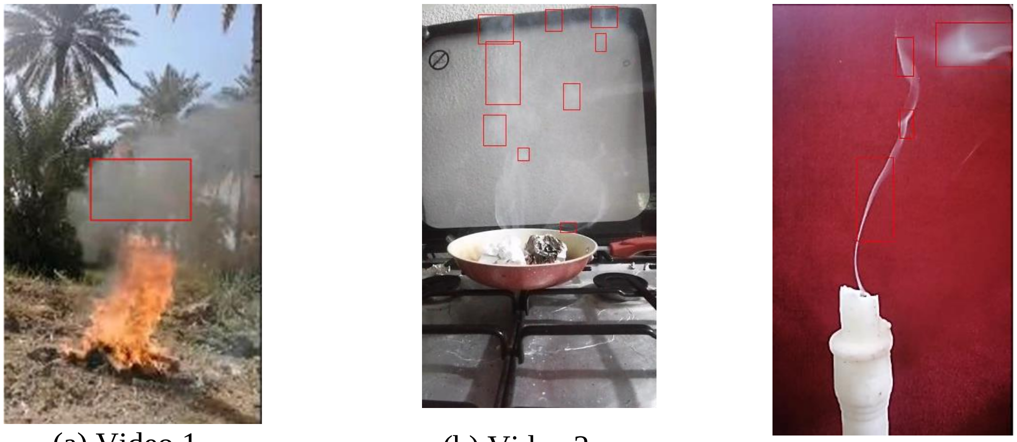
Table 4.1 Evaluation of the real-time smoke detection for indoor and outdoor.

Video name	Number of frames	N_S	True Positive	True Negative	False Positive	False Negative	Accuracy %
Video 1	45	6	5	0	0	1	83.3
Video 2	30	4	4	0	0	0	100
Video 3	64	8	6	0	2	0	75
Video 4	50	6	6	0	0	0	100
Video 5	70	9	9	0	0	0	100
Video 6	84	11	11	0	0	0	100
Total	**343**	**44**	**41**	**0**	**2**	**1**	**93.1**
Total indoor	**75**	**10**	**9**	**0**	**0**	**1**	**90**
Total outdoor	**268**	**34**	**32**	**0**	**2**	**0**	**94.1**

*Note: N_s the number of the selected, tested frame (see Section 3.6)

4.3.2 Offline Smoke Detection Results

Offline tests of our recorded video smoke detection for various indoor and outdoor places are presented in Figure 4.2. Three different smoke videos in different regions are used in the tests. The videos show smoke in various colors, ranging from gray to white, as bound in the red boxes for each video shown in Figure 4.2. Table 4.2 summarizes the real-time experiment results. The average accuracy of the smoke detection system for all three videos computed according to Equation (2.37) is 92.1%.



(a) Video 1

(b) Video 2

V

(c) Video 3

Figure 4.2 Test results of the offline smoke detection for (a) outdoor. (b and c) outdoor.

Table 4.2 Evaluation of the offline smoke detection for indoor and outdoor

Video name	Number of frames	N_S	True Positive	True Negative	False Positive	False Negative	Accuracy %
Video1	30	4	4	0	0	0	100
Video2	120	15	14	0	1	0	93.3
Video3	150	19	17	0	2	0	89.4
Total	**300**	**38**	**35**	**0**	**3**	**0**	**92.1**

*Note: N_s the number of the selected, tested frame (see Section 3.6)

The suggested method is compared with previous smoke detection systems based on color and other characteristics for the same used dataset in this work. The method is compared with the Yang et al. (Jia et al. 2016) method, which depends on using *CIE L*a*b** color space, optical flow and other characteristics where the precision equals 93% for eleven video databases from KMU Fire & Smoke Database and VisiFire (A.E. Çetin 2014; KMU Fire & Smoke Database 2012), as shown in Table 4.3. The videos features vary smoke colors, shapes, and speeds in indoor and outdoor places, as depicted in Figure 4.3. The proposed method's precision from Equation (2.36) reaches up to 96.3% as shown in Table 4.4. The higher precision means a lower false alarm. As a result of the poor video quality, videos 6 and 7 had lower precision. The precision of the forest videos is nearly 99% in videos 9, 10, and 11 for both methods (this papers and Yang’s method). These improved results are mainly due to using multi-thresholds for smoke color detection.

No.	Description
Video1	Smoke spread quickly across the field.
Video2	Short distance indoor rapid spread of cotton smoke.
Video3	Smoke spreads quickly across a field while a person is walking.
Video4	At a relatively short distance, indoor smoke diffuses slowly.
Video5	Smoke from a pretty near distance
Video6	There is smoke in the sky not very far away.
Video7	Indoor leaf smoke quickly dissipates over a small area.
Video8	rapid outdoor smoke dispersal at close range.
Video9	rapid smoke dispersion over a distant low hill.
Video10	rapid smoke dispersion over a distant low hill.
Video11	Long distances slow smoke dispersion on low hills.

Table 4.3 The description of videos.

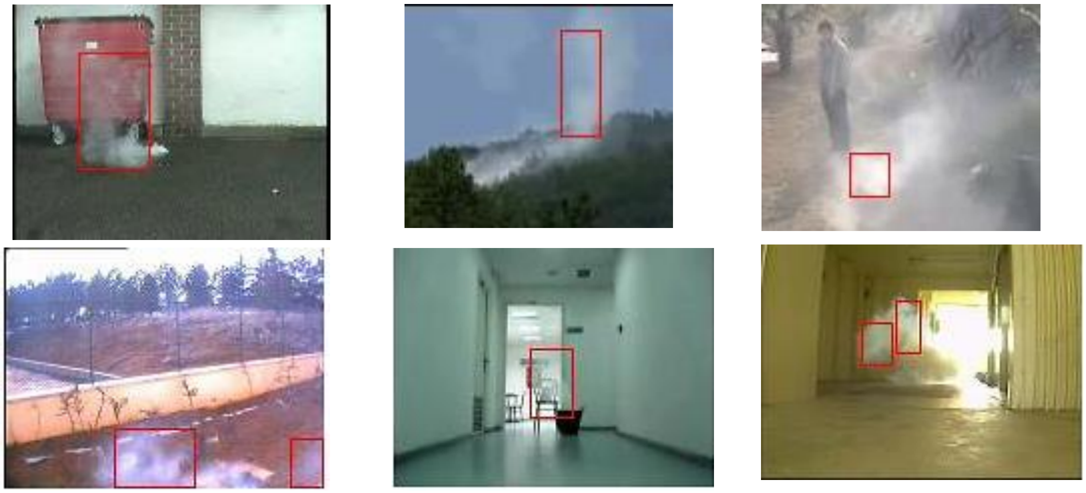


Figure 4.3 The proposed smoke detection using KMU Fire & Smoke Database and VisiFire dataset.

Table 4.4 Results for smoke detection performance comparison.

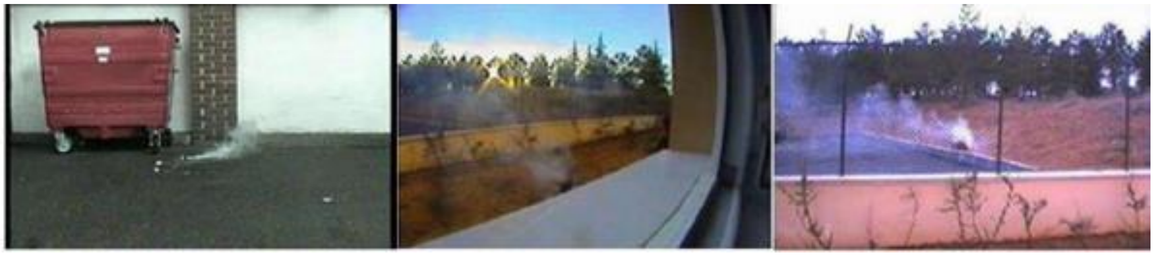
Video name	Number of frames	N_S	True Positive	False Positive	Precision %
Video1	80	10	8	2	80
Video2	1446	181	179	1	99
Video3	2875	359	355	1	99
Video4	480	60	59	0	100
Video5	896	112	100	3	97
Video6	3176	397	345	10	97
Video7	624	78	70	7	90
Video8	6084	761	750	9	98
Video9	2328	291	288	1	99
Video10	7624	953	366	0	100
Video11	2888	361	360	0	100
Total	**28501**	**3563**	**2880**	**34**	**96.3**

The speed of detecting smoke and recognizing it as a dangerous element in camera footage is crucial. Table 4.5 compares the proposed techniques with state-of-the-art methods (Gagliardi, de Gioia, and Saponara 2021; Gagliardi and Saponara 2020; Toreyin, Dedeoglu, and Cetin 2006; Yu, Mei, and Zhang 2013b) using the same test video in Figure 4.5. The comparison measures the delay in the number of frames to detect smoke. As a result, the new method taken in this study is better suited to early detection than the others.

Table 4.5 Comparison of detection delay in terms of the number of frames with the state-of-the-art.

Videos	Duration (in frames)	N_s	Delay in smoke detection (in frames)				
			(Toreyin, Dedeoglu, and Cetin 2006)	(Yu, Mei, and Zhang 2013b)	(Gagliardi and Saponara 2020)	(Gagliardi, de Gioia, and Saponara 2021)	used method
Video 1	900	113	98	86	9	11	2
Video 2	244	31	127	121	19	21	3
Video 3	630	79	132	118	120	39	5

*Note: N_s the number of the selected, tested frame (see Section 3.6)



Video 1 Video 2 Video 3

Figure 4.4 Videos used for comparison with state-of-art smoke detection techniques.

4.4 Results of Flame Detection based on Non-Adaptive Thresholds

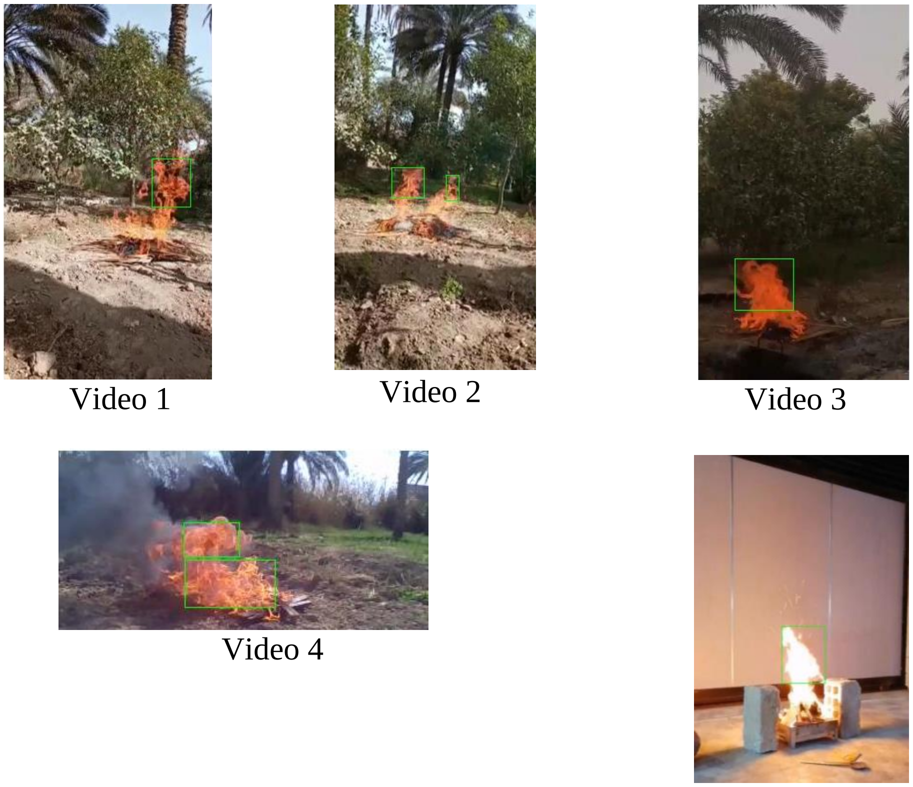
Flame detection results for the outdoor environment are categorized into two types: real-time results and recorded video results.

4.4.1 Real-time Flame Detection Results

Real-time tests of flame detection for various outdoor places are presented in Figure 4.5. Five different flame videos in different regions are used in the tests. The videos depict flames in various places, each with unique external factors. The detected flames are enclosed within green boxes. A summary of the real-time experimental results are be found in Table 4.6. The average accuracy of the smoke detection system for all five videos computed according to Equation (2.37) is 94.4%.

The proposed system on Figure 4.5 (*Video 1)* and (*Video 2)* demonstrate the system's ability to distinguish the color of the flame from the color of sunlight. Furthermore, the system can detect different flame colors, as seen in Figure 4.5 (*Video 3* and *Video 5*).

The timely identification of fires is of utmost importance, and the system being proposed effectively addresses this challenge by swiftly detecting flame in real-time. By integrating the *LWT* into the system, the response time has been significantly improved to less than 0.12 seconds. This enhancement has led to a substantial reduction in the overall time required for flame detection, effectively reducing it by more than half from the previous duration of 0.24 seconds.



Video 1 Video 2 Video 3

Video 4

Video 5

Figure 4.5 Test results of the real-time non-adaptive flame detection for outdoors.

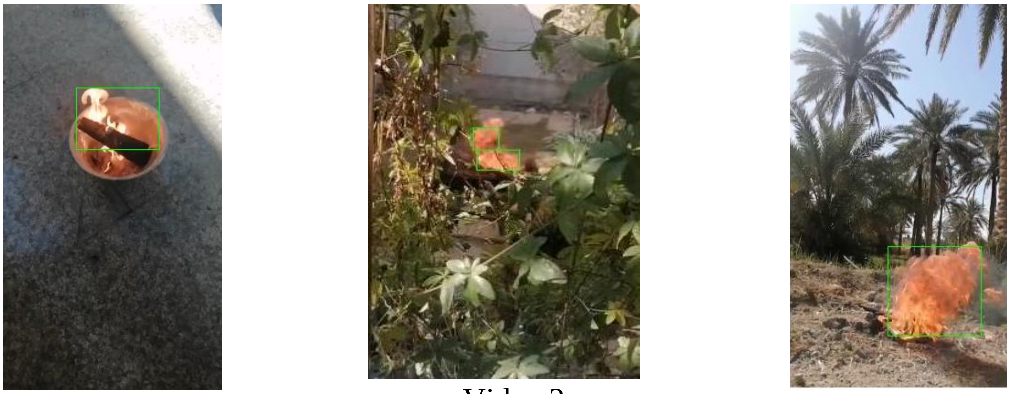
Table 4.6 Evaluation of the real-time non-adaptive flame detection for outdoors.

Video name	Number of frames	N_S	True Positive	True Negative	False Positive	False Negative	Accuracy %
Video 1	49	6	5	0	0	1	83.3
Video 2	97	12	11	0	1	0	91.6
Video 3	48	6	6	0	0	0	100
Video 4	30	4	4	0	0	0	100
Video 5	67	8	8	0	0	0	100
Total	**291**	**36**	**34**	**0**	**1**	**1**	**94.4**

*Note: N_s the number of the selected, tested frame (see Section 3.6)

4.4.2 Offline Flame Detection Results

Offline tests of our recorded video flame detection for various outdoor places are presented in Figure 4.6. Three different flame videos in different regions are used in these tests. The videos depict flames in various places, each with unique external factors. The detected flames are enclosed within green boxes. Table 4.7 summarizes the offline experiment results. The average accuracy of the flame detection system for all three videos computed according to Equation (2.37) is 94.8%.



Video 1 Video 2 Video 3

Figure 4.6 Test results of the offline non-adaptive flame detection for outdoors.

Table 4.7 Evaluation of the offline non-adaptive flame detection for outdoors.

Video name	Number of frames	N_S	True Positive	True Negative	False Positive	False Negative	Accuracy %
Video1	390	49	47	0	1	1	95.9
Video2	330	41	37	0	1	3	90.2
Video3	210	26	26	0	0	0	100
Total	**930**	**116**	**110**	**0**	**2**	**4**	**94.8**

*Note: N_s the number of the selected, tested frame (see Section 3.6)

The system's performance is evaluated using the datasets KMU Fire & Smoke Database and VisiFire (A.E. Çetin 2014; KMU Fire & Smoke Database 2012). The datasets have various resolutions, comprising diverse scenes, fire environments, and backgrounds. This heterogeneity in the environment and background provides an opportunity to effectively assess the system's performance. The details of the videos in the dataset are presented in

Table 4.8. In order to assess how well the proposed method performs on this dataset, a thorough empirical evaluation was conducted, using a total of 7100 frames sourced from 15 positive and negative video clips found on the internet. The frame rate of the video data ranges from 20 to 25 frames per second, and the image resolution is 320 × 240 and 400 × 256. A sample of these videos is depicted in Figure 4.7.

Table 4.8 The specification videos used for testing the non-adaptive flame detection

Video sequence	Total frames	Fire frames	Non-Fire frames	Video description
Video 1	439	433	6	Barbeq
Video 2	260	260	0	Controlled Environment 1 320 × 240
Video 3	246	246	0	Controlled Environment 2 320 × 240
Video 4	208	208	0	Controlled Environment 3 320 × 240
Video 5	1201	1070	131	Backyard
Video 6	708	708	0	Fire 1 400 × 256
Video 7	200	200	0	Forest 1 400 × 256
Video 8	245	245	0	Forest 2 400 × 256
Video 9	255	255	0	Forest 3 400 × 256
Video 10	219	219	0	Forest 4 400 × 256
Video 11	216	216	0	Forest 5 400 × 256
Video 12	218	218	0	Forest 6 400 × 256
Video 13	789	625	164	Highway 640 × 360
Video 14	1201	1129	72	Field 320 × 240
Video 15	402	402	0	Farm 320 × 240
Video 16	357	0	357	Fire moving color car 320 × 240
Video 17	306	0	306	Person with fire colored shirt 320 × 240
Total	7470	6434	1036	

The first 15 videos contain a multi-environment fire, while the reminder videos contain moving items with fire-like regions to evaluate the accuracy of the proposed method. High detection rates and low false alarms are crucial for thoroughly evaluating the proposed method.

Two different groups of evaluation metrics are employed to assess the performance of each method comprehensively. The first set of metrics, which includes false negatives, false positives (also referred to as false alarm rates), and accuracy from Equation (2.37), is used to compare the proposed system with related work (T. C. Chen, Wu, and Chiou 2004; X. F. Han et al. 2017; Khalil et al. 2021; Shidik et al. 2013). The second set of metrics, precision and recall, from Equations (2.36) and (2.35), is used when the cost of false positives is very high. Table 4.8 compares the proposed method's accuracy, true negative rate, and false negative rate with state-of-the-art methods.

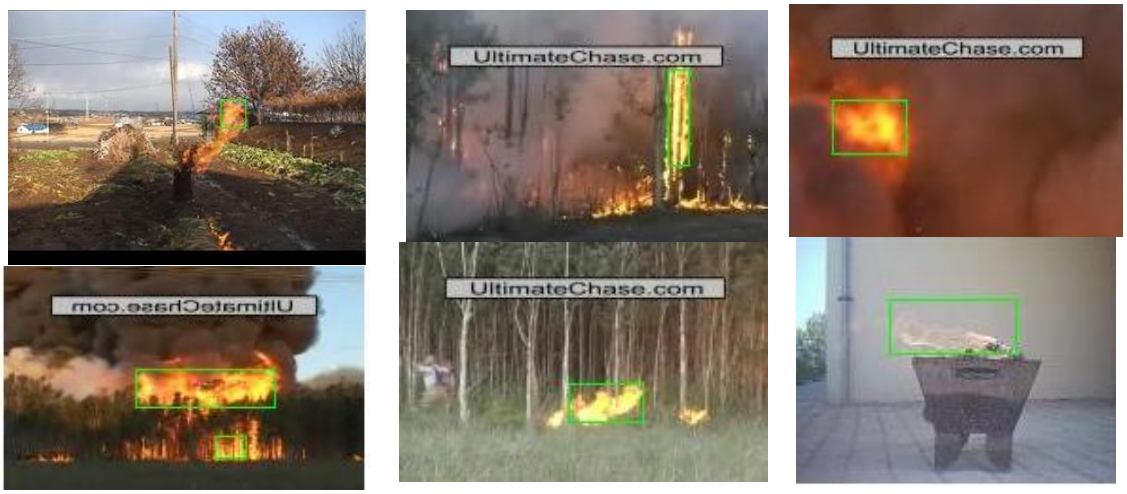


Figure 4.7 The proposed non-adaptive flame detection using KMU and VisiFire datasets.

The experimental results demonstrate that the method's accuracy about 98.22%, indicating that the proposed method has good accuracy and performs well in various settings. In order to assess the effectiveness of the suggested approach more comprehensively, standard performance evaluation metrics such as precision and recall are employed. As seen in Table 4.9, the recall and precision of the proposed method exhibit the most favorable performance compared to existing methods, except for the recall in the Khalil method (Khalil et al. 2021). Based on the experimental results, our proposed method demonstrates high accuracy and stability, with a correct rate of approximately 98%. It has been confirmed that our new approach exceeds previous methods in precision. Nevertheless, it is important to note that our algorithms have limitations, and low-quality videos may result in a false negative.

Table 4.9 Comparison of the proposed system with the related work

Refrences	True Positive	True Negative	False Positive	False Negative	Accuracy %
(T. C. Chen, Wu, and Chiou 2004)	5791	643	382	746	85.08
(Shidik et al. 2013)	5167	1267	347	791	78.68
(X. F. Han et al. 2017)	6278	189	431	697	92.59
(Khalil et al. 2021)	6293	137	1087	41	97.42
proposed work	**790**	**15**	**42**	**86.75**	**98.22**

Table 4.10 Comparison of the proposed system with the related work in terms of recall and precision

Refrencse	Recall	Precision
(T. C. Chen, Wu, and Chiou 2004)	0.8859	0.9381
(Shidik et al. 2013)	0.8672	0.9371
(X. F. Han et al. 2017)	0.9001	0.9358

(Khalil et al. 2021)	0.9935	0.8527
proposed work	**0.9015**	**0.9495**

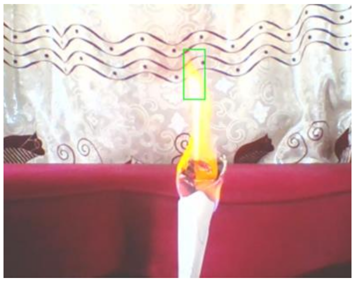
4.5 Results of Flame Detection based on Adaptive Thresholds

Flame detection results for indoor and outdoor environments are categorized into real-time and recorded video results.

4.5.1 Real-time Flame Detection Results

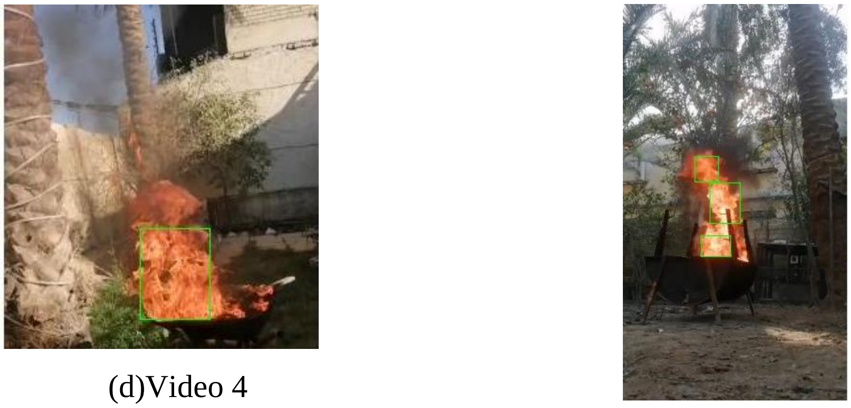
Real-time tests of flame detection for various indoor and outdoor places are presented in Figure 4.8. Five different flame videos in different regions are used in the tests. The videos depict flames with different backgrounds and external influences, and the detected flames are highlighted with green boxes. The real-time experiment results are summarized in Tables 4.11 and 4.12. These tables included flame detection average accuracy results according to Equation (2.37) using Otsu and Kapur objective functions-based *PSO* and *FOA* criteria.

(a)Video 1



(b)Video 2

(c)Video 3



(d)Video 4

(e)Video 5

Figure 4.8 Test results of the real-time adaptive flame detection for (a, b, and c) indoor. (d and e) outdoor.

Table 4.11 Evaluation of adaptive flame detection in real-time by PSO with objective functions (Outs and Kapur).

Video name	Frame number	N_s	Otsu					Kapur				
			TP	TN	FP	FN	ACC%	TP	TN	FP	FN	ACC%
Video 1	246	49	48	0	1	0	97.9	48	0	1	0	97.9
Video 2	270	23	23	0	0	0	100	23	0	0	0	100
Video 3	330	41	40	0	1	0	97.5	40	0	0	1	97.5
Video 4	390	31	30	0	0	1	96.7	29	0	2	0	93.5
Video 5	640	80	74	0	2	4	92.5	73	0	3	4	91.2
Total	**1876**	**224**	**215**	**0**	**4**	**5**	**95.9**	**213**	**0**	**6**	**5**	**95**
Total indoor	**846**	**113**	**111**	**0**	**2**	**0**	**98.2**	**111**	**0**	**1**	**1**	**98.2**
Total outdoor	**1030**	**111**	**104**	**0**	**2**	**5**	**93.6**	**102**	**0**	**5**	**4**	**91.8**

*Note: N_s the number of the selected, tested frame (see Section 3.6)

Table 4.12 Evaluation of adaptive flame detection in real-time by FOA with objective functions (Outs and Kapur)

Video Name	Frame number	N_s	Otsu					Kapur				
			TP	TN	FP	FN	ACC%	TP	TN	FP	FN	ACC%
Video 1	246	49	48	0	1	0	97.9	46	0	2	1	100
Video 2	270	23	23	0	0	0	100	22	0	0	1	95.6
Video 3	330	41	40	0	1	0	97.5	40	0	0	1	97.5
Video 4	390	31	30	0	0	1	96.7	28	0	1	2	84.8
Video 5	640	80	74	0	2	4	92.5	73	0	3	4	91.2
Total	**1876**	**224**	**215**	**0**	**4**	**5**	**95.9**	**209**	**0**	**6**	**9**	**93.3**
Total indoor	**846**	**113**	**111**	**0**	**2**	**0**	**98.2**	**108**	**0**	**2**	**3**	**95.5**
Total outdoor	**1030**	**111**	**104**	**0**	**2**	**5**	**93.6**	**101**	**0**	**4**	**6**	**90.9**

*Note: N_s the number of the selected, tested frame (see Section 3.6)

Tables 4.11 and 4.12 show that the Otsu algorithm gives a better threshold for flame detection than the Kapur algorithm. The flame detection system's accuracy (ACC) about 96% for Otsu and 94% for Kapur.

The reason for outperforming the Otsu results of Kapur in fire detection is the method for calculating the thresholds. First, Otsu's method is based on maximizing the between-class variance, which means it tries to find a threshold that maximizes the difference between the foreground and background regions. This approach is particularly suitable for fire detection because fires typically contrast strongly with their surroundings, making it easy to differentiate them from the background. On the other hand, Kapur's method is based on the concept of entropy, which tries to find a threshold that maximizes the entropy of the foreground and background regions, where a uniform distribution of intensity fire values.

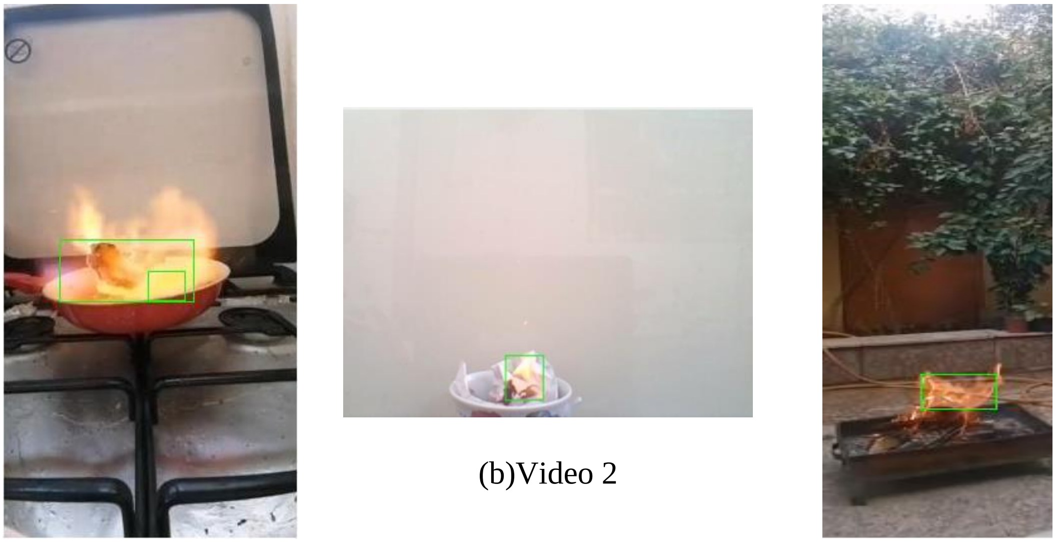
The average accuracy of indoor flame detection beats the average outdoor accuracy because the flame seems more visible in low light. The system's average accuracy of the indoors is 97.5%. In comparison, the average outdoor accuracy about 92.4%.

The tests for videos such as Figure 4.8 (*Video 4)* and (*Video 5)* prove the ability of the proposed system to distinguish the color of the flame from the color of sunlight. Also, the system can ignore the effect of lights similar to fire, as shown in the test video (*Video 1)*. Furthermore, the system can detect the small size of a flame, as seen in the test video (*Video 3)*.

The timely identification of fires is of utmost importance, and the system being proposed effectively addresses this challenge by swiftly detecting flame in real-time. The integration of the *LWT* into the system has resulted in a significant improvement in response time, reducing it to less than 0.6 seconds. This enhancement has led to a substantial reduction in the overall time required for flame detection, effectively reducing it by more than half from the previous duration of 1.2 seconds.

4.5.2 Offline Flame Detection Results

Offline tests of our recorded video flame detection for various indoor and outdoor places are presented in Figure 4.9. Three different flame videos in different regions are used in the tests. The videos depict flames in various places, each with unique external factors. The detected flames are enclosed within green boxes. Table 4.13 summarizes the offline experiment results. The average accuracy of the flame detection system for all three videos computed according to Equation (2.37) for each one is above 96%. Also, the Otsu algorithm gives a better threshold for flame detection than the Kapur algorithm. The flame detection system's accuracy is more than 97% for Otsu and 96% for Kapur.



(a)Video 1

(b)Video 2

(c)Video 3

Figure 4.9 Test results of the offline adaptive flame detection for (a and b) indoor. (c) outdoor.

Table 4.13 Evaluation of the offline adaptive flame detection for indoor and outdoor.

Methods	Video name	Number of frames	N_S	TP	TN	FP	FN	Accuracy %

Otsu-PSO	Video1	60	8	8	0	0	0	100
	Video2	630	79	78	0	0	1	98.7
	Video3	330	41	39	0	0	2	95.1
	Total	**1020**	**128**	**125**	**0**	**0**	**3**	**97.7**
Otsu-FOA	Video1	60	8	8	0	0	0	100
	Video2	630	79	78	0	0	1	98.7
	Video3	330	41	38	0	1	2	92.6
	Total	**1020**	**128**	**124**	**0**	**1**	**3**	**96.8**
Kapur-PSO	Video1	60	8	8	0	0	0	100
	Video2	630	79	77	0	1	1	97.5
	Video3	330	41	38	0	1	2	92.8
	Total	**1020**	**128**	**123**	**0**	**2**	**3**	**96**
Kapur-FOA	Video1	60	8	7	0	1	0	87.5
	Video2	630	79	77	0	1	1	97.5
	Video3	330	41	39	0	2	0	95.1
	Total	**1020**	**128**	**123**	**0**	**4**	**1**	**96**

*Note: N_s the number of the selected, tested frame (see Section 3.6)

The system's performance is evaluated using the datasets FireSense (Grammalidis, Dimitropoulos, and Cetin 2017) and VisiFire (A.E. Çetin 2014). The datasets have various resolutions, comprising diverse scenes, fire environments, and backgrounds. This heterogeneity in the environment and background provides an opportunity to effectively assess the system's performance. The details of the videos in the dataset are presented in Table 4.14. The first 11 videos containing a firing frame are collected from VisiFire (A.E. Çetin 2014), while the others have non-fire videos from FireSense (Grammalidis, Dimitropoulos, and Cetin 2017). A sample of these videos is depicted in Figure 4.10.

Table 4.14 The specification that utilized to evaluate the proposed adaptive flame detection.

Video	Frame rate	Frames	Resolution	Fire	Notes
Vid 1	15	260	400 × 256	Yes	Controlled1
Vid 2	15	246	400 × 256	Yes	Controlled2
Vid 3	15	208	400 × 256	Yes	Controlled3
Vid 4	15	200	400 × 256	Yes	Forest1
Vid 5	15	245	400 × 256	Yes	Forest2
Vid 6	15	255	400 × 256	Yes	Forest3
Vid 7	15	219	400 × 256	Yes	Forest4
Vid 8	15	216	400 × 256	Yes	Forest5
Vid 9	2	241	320 × 240	Yes	fBackYardFire
Vid 10	5	236	320 × 240	Yes	Fire1
Vid 11	29.97	140	320 × 240	Yes	40m PanFire 20060824
Vid 12	10	155	320 × 240	No	negsVideo2.859
Vid 13	10	160	320 × 240	No	negsVideo3.860
Vid 14	30	439	480 × 368	No	negsVideo5.862
Vid 15	30	541	640 × 368	No	negsVideo7.864
Vid 16	25	645	640 × 480	No	negsVideo9.866
Vid 17	25	246	360 × 288	No	negsVideo10.1072
Vid 18	25	180	320 × 240	No	negsVideo11.1073
Vid 19	24.46	272	352 × 288	No	negsVideo13.1075
Vid 20	18.51	196	1600 × 1200	No	negsVideo16.1077

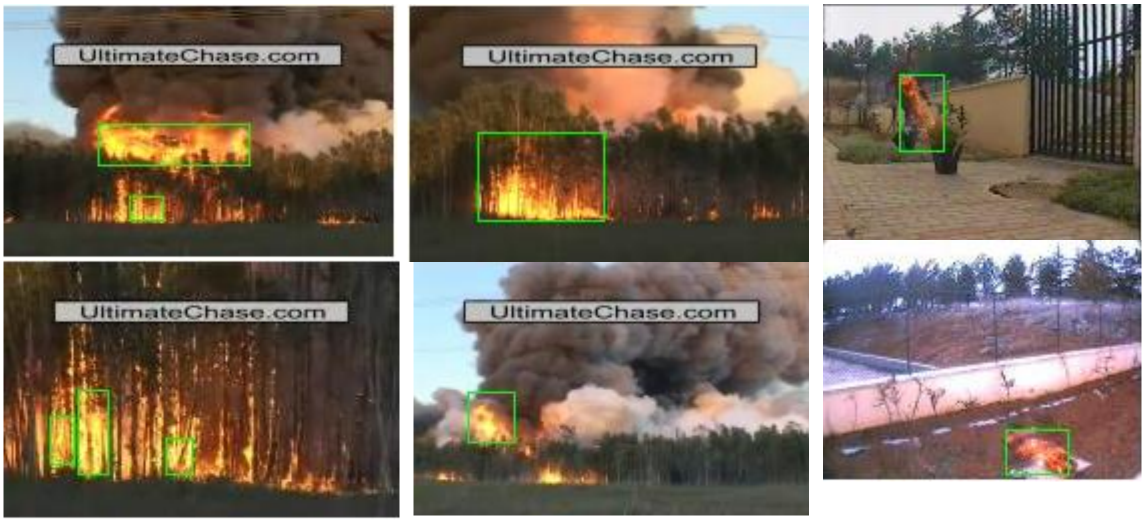


Figure 4.10 The proposed adaptive flame detection using VisiFire datasets.

During the experiment, several metrics were used to evaluate the performance of the detection system. These metrics included the true-positive rate (*TPR) or* recall from Equation (2.35), true-negative rate *TNR*, false-positive rate (*FPR)*, and false-negative rate (*FNR)*. The (*TPR)* is determined by dividing the number of correctly detected fire frames by the total number of video frames as well as the overall number of flame-filled frames. Similarly, the (*TNR)* is calculated by dividing the number of correctly detected non-fire frames by the total number of video frames as well as the overall number of non-flame-filled frames. On the other hand, the (*FPR)* is the ratio of false positives (instances that are mistakenly classified as positive) to the total number of actual negatives. It measures the proportion of negative instances that are incorrectly classified as positive. Lastly, the (*FNR)* is the ratio of false negatives (instances that are mistakenly classified as negative) to the total number of actual positives. It measures the proportion of positive instances that are incorrectly classified as negative.

Two different groups of evaluation metrics are employed to assess the performance of each method comprehensively. The first set of metrics, which includes *TPR* and *FNR*, are used to compare the proposed system with related work (Ko, Cheong, and Nam 2009; Töreyin et al. 2006; Truong and Kim 2012; Wahyono et al. 2022). The related work is named as Meth1, Meth2, Meth3, and Meth4, respectively. Table 4.15 compares the *TPR* and *FNR* of the proposed method with state-of-the-art methods for Otsu and Kapur objective functions-based *PSO* and *FOA* criteria. The second set of metrics, which includes *TNR* and *FPR*, compares the proposed system with related work (Wahyono et al. 2022). Table 4.16 compare the *TNR* and *FPR* of the proposed method with state-of-the-art methods for Otsu and Kapur objective functions-based *PSO* and *FOA* criteria.

Unfortunately, we could not compare our suggested approach to other methods for all testing videos due to limited resources and the difficulties of doing additional research. Instead, we compared them using the movie they used in their experiment. For video testing, 11 VisiFire video were utilized, and the results were compared with Wahyono (Wahyono et al. 2022). The *TPR* for our methods outperforms the Wahyono method, as shown in Table 4.15. Furthermore, compared with three videos, our methods beat Töreyin (Töreyin et al. 2006) and Ko (Ko, Cheong, and Nam 2009).

Table 4.15 Results of the VisiFire dataset's TPR and FNR comparison, expressed as a percentage

Video method		Vid 1	Vid 2	Vid 3	Vid 4	Vid 5	Vid 6	Vid 7	Vid 8	Vid 9	Vid 10	Vid 11	Avg
Meth1	TPR	34	87.5	73.7	-	-	-	-	-	-	-	-	65.1
	FNR	0	4.9	10	-	-	-	-	-	-	-	-	4.96
Meth2	TPR	55.2	77.7	97.7	-	-	-	-	-	-	-	-	76.9
	FNR	2	0	0	-	-	-	-	-	-	-	-	0.66
Meth3	TPR	94.9	-	95	-	-	-	-	-	-	-	-	94.9
	FNR	5.02	-	5	-	-	-	-	-	-	-	-	5.01
Meth4	TPR	100	100	100	100	100	92.2	100	98.6	98.6	74.8	88.8	95.7
	FNR	0	0	0	0	0	7.8	0	1.4	1.4	25.2	11.2	4.3
POS-Otsu	TPR	100	100	100	100	100	95.4	100	98.9	100	73.9	94.8	**96.6**
	FNR	0	0	0	0	0	4.6	0	1.1	0	26.1	5.2	**3.4**
POS-Kapur	TPR	100	100	100	100	100	93.4	100	98.8	100	73.8	92.4	**96.2**
	FNR	0	0	0	0	0	6.6	0	1.2	0	26.2	7.6	**3.8**
FOA-Otsu	TPR	100	100	100	100	100	95.4	100	98.9	100	73.9	94.8	**96.6**
	FNR	0	0	0	0	0	4.6	0	1.1	0	26.1	5.2	**3.4**
FOA-Kapur	TPR	100	100	100	100	100	93	100	98.7	100	73.7	92.4	**96.2**
	FNR	0	0	0	0	0	7	0	1.3	0	26.3	7.6	**3.8**

Also, it is important to test how well the method works with videos without fire. In this regard, the proposed method is evaluated using FireSense videos and compared with the Wahyono method (Wahyono et al. 2022). The presented results in Table 4.16 demonstrate that our proposed method has achieved a *TNR* value about 96%, surpassing the *TNR* value of the Wahyono approach (Wahyono et al. 2022) of around 38.73%. The proposed system reduced the *FPR* to less than 3.5%, which achieved one of our objectives.

Table 4.16 Results of TNR and FPR comparison on the FireSense Dataset

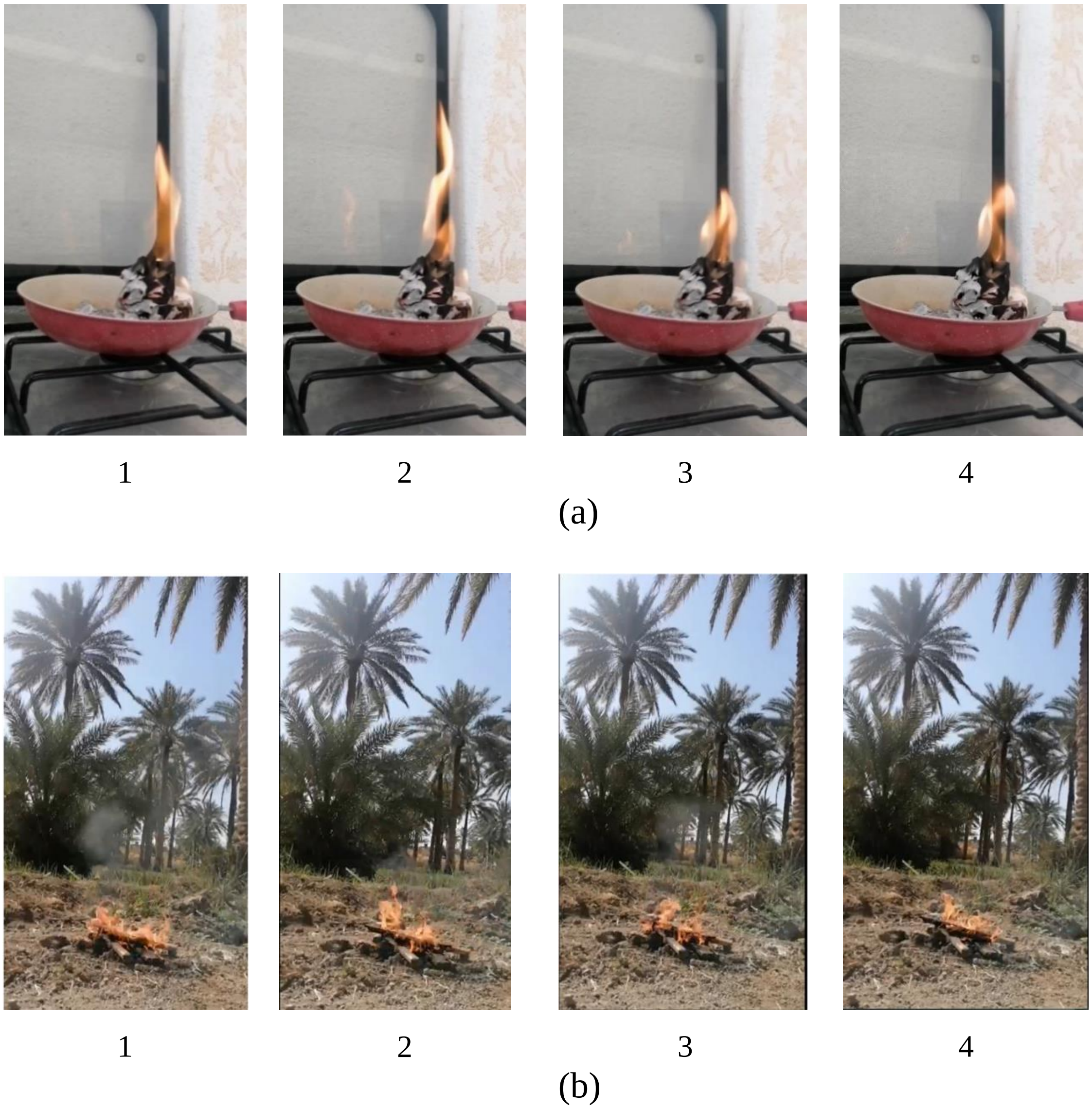
Video	Wahyono		Proposed methods							
			POS-Otsu		POS-Kapur		FOA-Otsu		FOA-Kapu	
	TNR	FPR	TNR	FPR	TNR	FPR	TNR	FPR	TNR	FPR
Vid 12	27.74	72.26	100	0	100	0	100	0	100	0
Vid 13	20.62	79.38	100	0	100	0	100	0	100	0
Vid 14	38.72	61.28	100	0	100	0	100	0	100	0
Vid 15	33.46	66.54	100	0	100	0	100	0	100	0
Vid 16	21.71	78.29	91.32	8.68	92.5	7.5	91.32	8.68	92.5	7.5
Vid 17	34.55	65.45	93.55	6.45	93.55	6.45	93.55	6.45	93.55	6.45
Vid 18	59.44	40.56	100	0	100	0	100	0	100	0
Vid 19	66.42	33.58	100	0	100	0	100	0	100	0
Vid 20	45.92	54.08	83.33	16.67	83.33	16.67	83.33	16.67	83.33	16.67
Avrg	**38.73**	**61.27**	**96.467**	**3.533**	**96.598**	**3.402**	**96.467**	**3.533**	**96.598**	**3.402**

4.6 Comparison between the Proposed methods of Flame Detection

This section compared the accuracy of the two flame detection methods from Equation (2.37). The first method depended on the static thresholds of the *HSV* color space, while the second method depended on the adaptive thresholds of the *HSV* color space. The test videos are shown in Figure 4.11. Table 4.17 shows the results for accuracy for the two method for indoor and outdoor places. For this proposed method, a total of 300 frames are captured for each video, and specific frames, namely the four frames, are chosen as an example.

From Table 4.17, the average accuracy of the indoor test video for static thresholds is low compared with adaptive thresholds test video. Furthermore, the Otsu algorithm gives better accuracy for flame detection than the Kapur algorithm.

Although the accuracy of the adaptive method is high, the time consumption for processing is higher than that of a non-adaptive method. The average time spent on fire detection for the adaptive method is 0.6 seconds, while the other method is 0.12 seconds.



1 2 3 4

(a)

1 2 3 4

(b)

Figure 4.11 The test video frames (a.) indoor (b.) outdoor

Table 4.17 Comparison between the static thresholds HSV and adaptive thresholds HSV

Frames	layars	HSV static threshold	HSV adaptive thresholds POS-Otsu			HSV adaptive thresholds POS-Kapur			HSV adaptive thresholds FOA-Otsu			HSV adaptive thresholds FOA-Kapur		
			thr$_1$	thr$_2$	thr$_3$	thr$_1$	thr$_2$	thr$_3$	thr$_1$	thr$_2$	thr$_3$	thr$_1$	thr$_2$	thr$_3$
indoor														
1	H	0, 0.2	0.15	0.35	0.52	0.17	0.36	0.53	0.17	0.35	0.52	0.17	0.34	0.52
	S	0.47, 0.98	0.17	0.35	0.51	0.17	0.36	0.53	0.16	0.34	0.51	0.18	0.35	0.53
	V	0.7, 0.98	0.17	0.35	0.52	0.17	0.36	0.52	0.17	0.34	0.51	0.18	0.36	0.54
2	H	0, 0.2	0.17	0.35	0.52	0.17	0.36	0.52	0.18	0.35	0.52	0.18	0.36	0.54
	S	0.47, 0.98	0.17	0.35	0.51	0.17	0.36	0.53	0.17	0.35	0.52	0.18	0.36	0.53
	V	0.7, 0.98	0.17	0.35	0.52	0.17	0.36	0.53	0.16	0.35	0.51	0.18	0.36	0.53
3	H	0, 0.2	0.17	0.35	0.52	0.17	0.36	0.53	0.17	0.34	0.51	0.18	0.35	0.53
	S	0.47, 0.98	0.18	0.34	0.51	0.17	0.36	0.53	0.17	0.35	0.51	0.18	0.36	0.53
	V	0.7, 0.98	0.17	0.35	0.52	0.17	0.36	0.52	0.18	0.35	0.52	0.18	0.36	0.53
4	H	0, 0.2	0.17	0.35	0.52	0.16	0.35	0.52	0.16	0.34	0.51	0.18	0.36	0.54
	S	0.47, 0.98	0.18	0.34	0.51	0.16	0.35	0.52	0.18	0.36	0.52	0.17	0.35	0.53
	V	0.7, 0.98	0.18	0.34	0.51	0.16	0.35	0.52	0.17	0.36	0.53	0.18	0.36	0.53
ACC		55%	97.5%			96.8%			97.3%			96.6%		
outdoor														
1	H	0, 0.2	0.19	0.36	0.52	0.16	0.35	0.53	0.19	0.36	0.52	0.16	0.34	0.53
	S	0.47, 0.98	0.18	0.35	0.51	0.17	0.38	0.54	0.19	0.35	0.52	0.18	0.35	0.54
	V	0.7, 0.98	0.18	0.34	0.50	0.17	0.38	0.54	0.19	0.35	0.51	0.18	0.36	0.54
2	H	0, 0.2	0.19	0.36	0.52	0.17	0.35	0.53	0.19	0.36	0.53	0.18	0.36	0.53
	S	0.47, 0.98	0.18	0.35	0.51	0.18	0.36	0.54	0.18	0.35	0.54	0.18	0.36	0.54
	V	0.7, 0.98	0.17	0.35	0.50	0.17	0.36	0.50	0.19	0.35	0.54	0.18	0.37	0.53
3	H	0, 0.2	0.17	0.36	0.52	0.17	0.36	0.53	0.18	0.35	0.53	0.18	0.36	0.54
	S	0.47, 0.98	0.16	0.35	0.51	0.18	0.35	0.54	0.19	0.35	0.53	0.18	0.36	0.54
	V	0.7, 0.98	0.18	0.35	0.50	0.16	0.35	0.55	0.17	0.35	0.52	0.19	0.37	0.54
4	H	0, 0.2	0.19	0.36	0.52	0.17	0.35	0.53	0.19	0.36	0.52	0.17	0.36	0.54
	S	0.47, 0.98	0.16	0.35	0.51	0.18	0.37	0.55	0.19	0.34	0.51	0.18	0.35	0.54

	V	0.7, 0.98	0.16	0.35	0.51	0.18	0.37	0.55	0.19	0.35	0.52	0.19	0.36	0.54
ACC		96.4%	98.2%			97.8%			98%			97.7%		

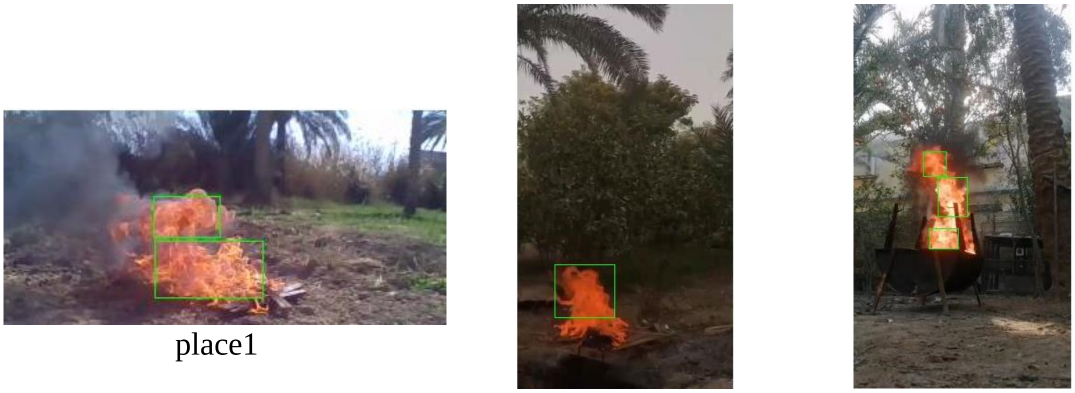
*thr represents the threshold

4.7 Real-World Fire Localization

	Actual Coordinates	Predicted Coordinates	Error (Meters)
Place 1	(2.5, 4.7, 0)	(2.3, 4.6, 0)	0.22
Place 2	(3, 3.4, 2.2)	(3.3, 3.1, 2)	0.47

Place 3	(4.3, 4, 1.5)	(3.8, 4.2, 1.3)	0.57

The proposed system's localization accuracy was evaluated in the final experiment by selecting three outdoor locations and employing the laptop camera for camera calibration and fire detection to determine the fire's real-world coordinates. The three different places of fire are shown in Figure 4.12, where the camera serves as a reference. Table 4.18 displays the results, which indicate an average localization error of 0.42 m. It can also be deduced from the table that the error increased as the test location moved further from the camera, although the localization error remained under 5 m.



place1

place 2

Place 3

Figure 4.12 Different videos for fire localization in three places.

Table 4.18 The actual and predicted coordinates of three random locations.

Chapter Five: Conclusions and Future Work

5.1 Conclusions

The proposed methods to detect the flame and smoke have used color detection in combination with motion detection as well as some morphological operation to enhance the detection results. The following conclusions can be depicted:

1. The proposed fire detection system combines color and motion detection approaches. It uses *RGB* video input, which is decomposed using *LWT* to reduce data processing while preserving fire features. The decomposed frames are then processing for color and motion detection to identify fire characteristics. Morphological post-processing removes unwanted pixels, and the detected fire area is calculated and bounded if it meets the threshold conditions.
2. The system employs three threshold types, including static multi-threshold for smoke detection, non-adaptive threshold for different color spaces, and adaptive threshold using optimization algorithms based on objective functions like Otsu and Kapur for flame detection.
3. The proposed approaches of fire detection are suitable for real-time indoor and outdoor areas and for multi-fire places with high detection rate accuracy.
4. Using the *Int-to-Int-HLWT* in preprocessing is reduces the processing time without effect to accuracy. The processing time is reduced about about 50% compared to the case without using *Int-to-Int-HLWT*.

5. The offline tests utilize the datasets, including KUM, VisiFire, and FireSense, as well as our recorded videos. The downloaded datasets include 92 fire videos, both smoke- and flame-based.
6. The offline test achieved about 92% average rate of the correct fire detection for the static multi-threshold smoke detection system. In contrast, the flame detection system-based non-adaptive threshold achieved about 94%. Moreover, the adaptive thresholds flame detection system achieved approximately 96%.
7. Online fire detection is tested in indoor and outdoor places. The indoor average accuracy was 90% for the static multi-threshold smoke detection system and 97.5% for the adaptive threshold flame detection system. The average outdoor fire detection accuracy for the static multi-threshold smoke detection system, non-adaptive, and adaptive thresholds flame detection systems was 94.1%, 94.4%, and 92.4%, respectively. The error of fire localization is achieved at less than 0.42 meters.
8. The proposed system can detect the smoke of the fire, for different densities of the smoke, even for very low densities of smoke under sunlight. Furthermore, it can detect smoke for gray and white backgrounds. This is because the combination of both color and motion detection.
9. The proposed systems can detect the flame of the fire under the sunlight and also for the background that is similar to the flame colors. The superiority of the proposed system for flame detection is because the color threshold used, which is based on optimization techniques.
10. The proposed approach of fire localization satisfies high identicality to the real word. The approach employs a projective transformation matrix

that has the ability to map between frame pixels and real-world positions.

11. The proposed adaptive fire detection system has significantly improved in reducing false positives compared to previous work. Implementing an optimized thresholding method has enabled the system to distinguish between a real fire and a non-fire more accurately, reducing false positives by 5%.

5.2 Future Work

To develop the performance of the present work, we suggest the following points of view for future work:

1. Extend the input of the proposed system to have a multi-camera instead single camera in order to increase the accuracy of fire localization in the real world.
2. Thermal camera can be merged with a digital camera for fire detection algorithms.
3. Replace the computer with an advanced microcontroller and the Internet of Things (*IoT*) that controls a complete fire detection and fighting system with the help of a special vehicle or drone.
4. Send firefighters a reliable fire alarm; cellular, Wi-Fi, and satellite networks can provide this alarm. Images of the scene, fire status, burning materials, and other information can be sent via communication channels and the fire alarm message. These informations help firefighters themselves prepare before arriving.

References

Alzughaibi, Arwa Darwish, Hanadi Ahmed Hakami, and Zenon Chaczko. 2015. “Review of Human Motion Detection Based on Background Subtraction Techniques.” *International Journal of Computer Applications* 122(13): 1–5.

Anutam, and Rajni. 2006. “COMPARATIVE ANALYSIS OF FILTERS AND WAVELET BASED THRESHOLDING METHODS FOR IMAGE DENOISING.” *Neurosurgery* 58(February): 13–21.

Araki, Tadashi et al. 2015. “Shape-Based Approach for Coronary Calcium Lesion Volume Measurement on Intravascular Ultrasound Imaging and Its Association with Carotid Intima-Media Thickness.” *Journal of Ultrasound in Medicine* 34(3): 469–82.

Aslam, Yasir, N. Santhi, N. Ramasamy, and K. Ramar. 2020. “A Review on Various Clustering Approaches for Image Segmentation.” In *Proceedings of the 4th International Conference on Inventive Systems and Control, ICISC 2020*, , 679–85.

Bamerni, Serwan Ali, and Ahmed Kh Al-Sulaifanie. 2019. “An Efficient Non-Separable Architecture for Haar Wavelet Transform with Lifting Structure.” *Microprocessors and Microsystems* 71: 102881.

https://doi.org/10.1016/j.micpro.2019.102881.

Bhandari, A. K., A. Kumar, and G. K. Singh. 2015. “Tsallis Entropy Based Multilevel Thresholding for Colored Satellite Image Segmentation Using Evolutionary Algorithms.” *Expert Systems with Applications* 42: 8707–30. http://dx.doi.org/10.1016/j.eswa.2015.07.025.

Bhandari, Ashish Kumar, Vineet Kumar Singh, Anil Kumar, and Girish Kumar Singh. 2014. “Cuckoo Search Algorithm and Wind Driven Optimization Based Study of Satellite Image Segmentation for Multilevel Thresholding Using Kapur’s Entropy.” *Expert Systems with Applications* 41(7): 3538–60. http://dx.doi.org/10.1016/j.eswa.2013.10.059.

“Calibration Patterns - MATLAB & Simulink.” 2023. https://www.mathworks.com/help/vision/ug/calibration-patterns.html (April 4, 2023).

Çelik, Turgay, and Hasan Demirel. 2009. “Fire Detection in Video Sequences Using a Generic Color Model.” *Fire Safety Journal* 44(2): 147–58.

Celik, Turgay, Hasan Demirel, Huseyin Ozkaramanli, and Mustafa Uyguroglu. 2007. “Fire Detection Using Statistical Color Model in Video Sequences.” *Journal of Visual Communication and Image Representation* 18(2): 176–85.

Çelik, Turgay, Hüseyin Özkaramanl, and Hasan Demirel. 2007.

“Fire and Smoke Detection without Sensors: Image Processing Based Approach.” In *15th European Signal Processing Conference (EUSIPCO 2007)*, Poznan, Poland: IEEE, 1794–98.

Çetin, A.E. 2014. “COMPUTER VISION BASED FIRE DETECTION SOFTWARE.” http://signal.ee.bilkent.edu.tr/VisiFire/Demo/SampleClips.html (December 20, 2022).

Çetin, A Enis et al. 2013. “Video Fire Detection – Review.” 23: 1827–43.

Chaki, Jyotismita, and Nilanjan Dey. 2019. *BEGINNER’S GUIDE TO IMAGE PREPROCESSING TECHNIQUES*. S.l.: CRC PRESS.

Chaki, Jyotismita, and Ranjan Parekh. 2011. “Plant Leaf Recognition Using Shape Based Features and Neural Network Classifiers.” *International Journal of Advanced Computer Science and Applications* 2(10).

Chandrasekaran, Koushik. 2021. “2D-Discrete Wavelet Transformation and Its Applications in Digital Image Processing Using MATLAB | by Koushik Chandrasekaran | Medium.” https://medium.com/@koushikc2000/2d-discrete-wavelet-transformation-and-its-applications-in-digital-image-processing-using-matlab-1f5c68672de3 (June 19, 2023).

Chen, Juan, Yaping He, and Jian Wang. 2010. “Multi-Feature

Fusion Based Fast Video Flame Detection." *Building and Environment* 45(5): 1113–22. http://dx.doi.org/10.1016/j.buildenv.2009.10.017.

Chen, Thou-ho Chao-ho, Ping-hsueh Wu, and Yung-chuen Chiou. 2004. "An Early Fire-Detection Method Based on Image Processing." In *2004 International Conference on Image Processing (ICIP)*, , 1707–10.

"Cluster Analysis and Clustering Algorithms - MATLAB & Simulink." 2022. https://ch.mathworks.com/discovery/cluster-analysis.html?s_tid=srchtitle_CLUSTER_1 (December 20, 2022).

Díaz-Cortés, Margarita Arimatea et al. 2018. "A Multi-Level Thresholding Method for Breast Thermograms Analysis Using Dragonfly Algorithm." *Infrared Physics and Technology* 93: 346–61. https://doi.org/10.1016/j.infrared.2018.08.007.

Eberhart, Russell, and James Kennedy. 1995. "New Optimizer Using Particle Swarm Theory." In *Proceedings of the International Symposium on Micro Machine and Human Science*, , 39–43.

Elavarasi, S. Anitha, J Akilandeswari, and B Sathiyabhama. 2011. "A Survey on Partition Clustering Algorithms." *International Journal of Enterprise Computing and Business Systems* 1(1): 1–14. http://www.ijecbs.com/January2011/N6Jan2011.pdf.

Ewees, Ahmed A., Mohamed Abd Elaziz, and Diego Oliva. 2018. “Image Segmentation via Multilevel Thresholding Using Hybrid Optimization Algorithms.” *Journal of Electronic Imaging* 27(6): 063008.

Farshi, Taymaz Rahkar, Recep Demirci, and Mohammad Reza Feizi-Derakhshi. 2018. “Image Clustering with Optimization Algorithms and Color Space.” *Entropy* 20(4): 296.

Foggia, Pasquale, Alessia Saggese, and Mario Vento. 2015. “Real-Time Fire Detection for Video-Surveillance Applications Using a Combination of Experts Based on Color, Shape, and Motion.” *IEEE Transactions on Circuits and Systems for Video Technology* 25(9): 1545–56.

Gagliardi, Alessio, Francesco de Gioia, and Sergio Saponara. 2021. “A Real-Time Video Smoke Detection Algorithm Based on Kalman Filter and CNN.” *Journal of Real-Time Image Processing* 18(6): 2085–95.

Gagliardi, Alessio, and Sergio Saponara. 2020. “Advised: Advanced Video Smoke Detection for Real-Time Measurements in Antifire Indoor and Outdoor Systems.” *Energies* 13(8): 18.

Ghaemi, Manizheh, and Mohammad Reza Feizi-Derakhshi. 2014. “Forest Optimization Algorithm.” *Expert Systems with Applications* 41(15): 6676–87. http://dx.doi.org/10.1016/j.eswa.2014.05.009.

Gong, Faming et al. 2019. “A Real-Time Fire Detection Method from Video with Multifeature Fusion.” *Computational Intelligence and Neuroscience* 2019: 18.

Gonzalez, Rafael C., Richard E. Woods, and S. L. Eddins. 2009. *Digital Image Processing ,Second Edition*. New Jersey: Parson.

Grammalidis, Nikos, Kosmas Dimitropoulos, and Enis Cetin. 2017. “FIRESENSE Database of Videos for Flame and Smoke Detection.” https://zenodo.org/record/836749 (February 10, 2023).

Günay, Osman, Kasim Taşdemir, B. Uğur Töreyin, and A. Enis Çetin. 2010. “Fire Detection in Video Using LMS Based Active Learning.” *Fire Technology* 46(3): 551–77.

Han, Dongil, and Byoungmoo Lee. 2009. “Flame and Smoke Detection Method for Early Real-Time Detection of a Tunnel Fire.” *Fire Safety Journal* 44(7): 951–61. http://dx.doi.org/10.1016/j.firesaf.2009.05.007.

Han, Xian Feng et al. 2017. “Video Fire Detection Based on Gaussian Mixture Model and Multi-Color Features.” *Signal, Image and Video Processing* 11: 1419–1425.

Harris, Chris, and Mike Stephens. 1998. “A COMBINED CORNER AND EDGE DETECTOR.” In *Proceedings of the 4th Alvey Vision Conference*, , 10–5244.

Horng, Ming Huwi. 2011. “Multilevel Thresholding Selection

Based on the Artificial Bee Colony Algorithm for Image Segmentation.” *Expert Systems with Applications* 38(11): 13785–91. http://dx.doi.org/10.1016/j.eswa.2011.04.180.

Horng, Ming Huwi, and Ting Wei Jiang. 2010. “Multilevel Image Thresholding Selection Based on the Firefly Algorithm.” In *In Proceedings of the 2010 7th International Conference on Ubiquitous Intelligence & Computing and 7th International Conference on Autonomic & Trusted Computing (UIC/ATC)*, , 58–63.

Husein, A. M. et al. 2019. “Motion Detect Application with Frame Difference Method on a Surveillance Camera.” *Journal of Physics: Conference Series* 1230(1): 012017.

“Image Processing Toolbox Documentation.” 2022. https://www.mathworks.com/help/images/ (December 28, 2022).

“Image Types in the Toolbox - MATLAB & Simulink.” 2022. https://www.mathworks.com/help/images/image-types-in-the-toolbox.html (December 28, 2022).

“Iraqi Ministry of Interior.” 2023. https://moi.gov.iq/?page=4417 (January 5, 2023).

Jia, Yang et al. 2016. “A Saliency-Based Method for Early Smoke Detection in Video Sequences.” *Fire Technology* 52(5): 1271–92.

Al Jumah, Abdullah. 2013. “Denoising of an Image Using Discrete Stationary Wavelet Transform and Various Thresholding Techniques.” *Journal of Signal and Information Processing* 04(01): 33–41.

Kapur, J.N., P.K. Sahoo, and A.K.C. Wong. 1985. “A New Method for Gray-Level Picture Thresholding Using the Entropy of the Histogram.” *COMPUTER VISION, GRAPHICS, AND IMAGE PROCESSING* 29(3): 273–85.

KC, Santosh, and Cholwich Nattee. 1970. “A Comprehensive Survey on On-Line Handwriting Recognition Technology and Its Real Application to the Nepalese Natural Handwriting.” *Kathmandu University Journal of Science, Engineering and Technology* 5(1): 31–55.

Khalil, Adnan et al. 2021. “Fire Detection Using Multi Color Space and Background Modeling.” *Fire Technology* 57: 1221–39. https://doi.org/10.1007/s10694-020-01030-9.

Kim, Sangdan. 2004. “Wavelet Analysis of Precipitation Variability in Northern California, U.S.A.” *KSCE Journal of Civil Engineering* 8(4): 471–77. http://link.springer.com/content/pdf/10.1007%2FBF02829169.pdf%5Cnfile:///Users/vaniarosa/Documents/Papers/Unknown/Unknown/Untitled-p2767.pdf%5Cnpapers://d28dd570-4187-41d3-8cf5-6edf67fea1b9/Paper/p2767.

"KMU Fire & Smoke Database." 2012. *KMU CVPR Lab*. https://cvpr.kmu.ac.kr/Dataset/Dataset.htm (March 10, 2023).

Ko, Byoung Chul, Kwang Ho Cheong, and Jae Yeal Nam. 2009. "Fire Detection Based on Vision Sensor and Support Vector Machines." *Fire Safety Journal* 44: 322–29.

Kolesov, I., P. Karasev, A. Tannenbaum, and E. Haber. 2010. "Fire and Smoke Detection in Video with Optimal Mass Transport Based Optical Flow and Neural Networks." In *Proceedings - International Conference on Image Processing, ICIP*, , 761–64.

Konrad, Janusz. 2009. The Essential Guide to Video Processing *Motion Detection and Estimation*. Second Edi. Elsevier Inc. http://dx.doi.org/10.1016/B978-0-12-374456-2.00001-3.

Lee, Byoungmoo, and Dongil Han. 2007. "Real-Time Fire Detection Using Camera Sequence Image in Tunnel Environment." In *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Springer, 1209–20.

León, Katherine, Domingo Mery, Franco Pedreschi, and Jorge León. 2006. "Color Measurement in L*a*b* Units from RGB Digital Images." *Food Research International* 39(10): 1084–91.

Li, Jiafu, Wenyan Tang, Jun Wang, and Xiaolin Zhang. 2019. "A Multilevel Color Image Thresholding Scheme Based on Minimum Cross Entropy and Alternating Direction Method of

Multipliers." *Optik* 183: 30–37. https://doi.org/10.1016/j.ijleo.2019.02.004.

Li, Yimang et al. 2023. "Real-Time Early Indoor Fire Detection and Localization on Embedded Platforms with Fully Convolutional One-Stage Object Detection." *Sustainability* 15(3): 1794.

Li, Youguo, and Haiyan Wu. 2012. "A Clustering Method Based on K-Means Algorithm." *Physics Procedia* 25: 1104–9. http://dx.doi.org/10.1016/j.phpro.2012.03.206.

Liu, Che-bin, and Narendra Ahuja. 2004. "Vision Based Fire Detection." *Proceedings of 17th Inernational Conference on Pattern Recognition (ICPR)* 4: 134–137.

Maitra, Madhubanti, and Amitava Chatterjee. 2008. "A Hybrid Cooperative-Comprehensive Learning Based PSO Algorithm for Image Segmentation Using Multilevel Thresholding." *Expert Systems with Applications* 34(2): 1341–50.

Mallat, Stephane G. 1989. "A Theory for Multiresolution Signal Decomposition: The Wavelet Representation." *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE* 11(7): 674–93.

Manikandan, S., K. Ramar, M. Willjuice Iruthayarajan, and K. G. Srinivasagan. 2014. "Multilevel Thresholding for Segmentation of Medical Brain Images Using Real Coded Genetic Algorithm." *Measurement: Journal of the International*

Measurement Confederation 47: 558–68.

Marques, Oge. 2011. *Practical Image and Video Processing Using MATLAB*. Hoboken: IEEE Press.

“Morphological Operations (Image Processing Toolbox).” 2022. http://www.ece.northwestern.edu/local-apps/matlabhelp/toolbox/images/morph13.html (December 29, 2022).

Nalley, D., J. Adamowski, and B. Khalil. 2012. “Using Discrete Wavelet Transforms to Analyze Trends in Streamflow and Precipitation in Quebec and Ontario (1954-2008).” *Journal of Hydrology* 475: 204–28. http://dx.doi.org/10.1016/j.jhydrol.2012.09.049.

Oliva, Diego et al. 2013. “Multilevel Thresholding Segmentation Based on Harmony Search Optimization.” *Journal of Applied Mathematics* 2013(575414).

OTSU, NOBUYUKI. 1979. “A Threshold Selection Method from Gray-Level Histograms.” *IEEE Transaction on Systems, Man and Cybernetics* (1): 62–66.

Pare, S., A. K. Bhandari, A. Kumar, and G. K. Singh. 2018. “A New Technique for Multilevel Color Image Thresholding Based on Modified Fuzzy Entropy and Lévy Flight Firefly Algorithm.” *Computers and Electrical Engineering* 70: 476–95. http://dx.doi.org/10.1016/j.compeleceng.2017.08.008.

Parida, Priyadarsan, and Nilamani Bhoi. 2017. “Wavelet Based Transition Region Extraction for Image Segmentation.” *Future Computing and Informatics Journal* 2(2): 65–78. https://doi.org/10.1016/j.fcij.2017.10.005.

Percival, Donald B. 2008. Nonlinear Time Series Analysis in the Geosciences *Analysis of Geophysical Time Series Using Discrete Wavelet Transforms: An Overview*.

Phillips, Walter III, Mubarak Shah, and Niels Da Vitoria Lobo. 2002. “Flame Recognition in Video.” *Pattern Recognition Letters* 23: 319–27.

Portes de Albuquerque, M., I. A. Esquef, A. R. Gesualdi Mello, and M. Portes de Albuquerque. 2004. “Image Thresholding Using Tsallis Entropy.” *Pattern Recognition Letters* 25(9): 1059–65.

Pradhan, BiswajeetAwang, Mohd Dini Hairi Bin Suliman, and Mohamad Arshad Bin Awang. 2007. “Forest Fire Susceptibility and Risk Mapping Using Remote Sensing and Geographical Information Systems (GIS).” *Disaster Prevention and Management: An International Journal* 16(3): 344–52.

Qi, Xiaojun, and Jessica Ebert. 2009. “A Computer Vision-Based Method for Fire Detection in Color Videos.” *International Journal of Imaging* 2(S09): 22–34.

Saponara, Sergio, Abdussalam Elhanashi, and Alessio Gagliardi. 2021. “Real-Time Video Fire/Smoke Detection Based on CNN in Antifire Surveillance Systems." *Journal of Real-Time Image Processing* 18: 889–900. https://doi.org/10.1007/s11554-020-01044-0.

Sathya, P. D., and R. Kayalvizhi. 2011a. "Modified Bacterial Foraging Algorithm Based Multilevel Thresholding for Image Segmentation." *Engineering Applications of Artificial Intelligence* 24(4): 595–615. http://dx.doi.org/10.1016/j.engappai.2010.12.001.

———. 2011b. "Optimal Multilevel Thresholding Using Bacterial Foraging Algorithm." *Expert Systems with Applications* 38(12): 15549–64. http://dx.doi.org/10.1016/j.eswa.2011.06.004.

Selvakumar, J., A. Lakshmi, and T. Arivoli. 2012. "Brain Tumor Segmentation and Its Area Calculation in Brain MR Images Using K-Mean Clustering and Fuzzy C-Mean Algorithm." In *IEEE-International Conference on Advances in Engineering, Science and Management (ICAESM-2012)*, , 186–90.

Shahadi, Haider Ismael, Razali Jidin, and Wong Hung Way. 2013. "High Performance FPGA Architecture for Dual Mode Processor of Integer Haar Lifting-Based Wavelet Transform." *International Review on Computers and Software* 8(9): 2058–67.

Shidik, Guruh Fajar et al. 2013. "Multi Color Feature, Background Subtraction and Time Frame Selection for Fire Detection." In

2013 International Conference on Robotics, Biomimetics, Intelligent Computational Systems., , 115–20.

Singla, Nishu. 2014. “Motion Detection Based on Frame Difference Method.” *International Journal of Information & Computation Technology* 4(15): 1559–65. http://www.ripublication.com/irph/ijict_spl/ijictv4n15spl_10.pdf.

Smith, Alvy Ray. 1978. “Color Gamut Transform Pairs.” In *SIGGRAPH 78 Conference Proceedings*, , 12–19.

Sturm, Peter F., and Stephen J. Maybank. 1999. “On Plane-Based Camera Calibration: A General Algorithm, Singularities, Applications.” In *IEEE Conference Computer Vision and Pattern Recognition*, , 432–37.

Sweldens, Wim. 1996. “The Lifting Scheme: A Custom-Design Construction of Biorthogonal Wavelets.” *Applied and Computational Harmonic Analysis* 3(2): 186–200.

Toreyin, B. Ugur, Yigithan Dedeoglu, and A. Enis Cetin. 2006. “Contour Based Smoke Detection in Video Using Wavelets.” In *14th European Signal Processing Conference*, , 1–5.

Töreyin, B. Uğur, Yiğithan Dedeoğlu, Uğur Güdükbay, and A. Enis Çetin. 2006. “Computer Vision Based Method for Real-Time Fire and Flame Detection.” *Pattern Recognition Letters* 27(1): 49–58.

Torrence, Christopher, and Gilbert P. Compo. 1998. "A Practical Guide to Wavelet Analysis." *Bulletin of the American Meteorological Society* 79(1): 61–78.

Truong, Tung Xuan, and Jong-Myon Kim. 2012. "Fire Flame Detection in Video Sequences Using Multi-Stage Pattern Recognition Techniques." *Engineering Applications of Artificial Intelligence* 25: 1365–72. http://dx.doi.org/10.1016/j.engappai.2012.05.007.

Tsai, Roger Y. 1987. "A Versatile Camera Calibration Technique for High_Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses." *IEEE J. Robotics and Automation* 3(4): 323–44.

"Understanding Color Spaces and Color Space Conversion - MATLAB & Simulink - MathWorks Switzerland." 2022. https://ch.mathworks.com/help/images/understanding-color-spaces-and-color-space-conversion.html?searchHighlight=color space&s_tid=srchtitle_color%2520space_1 (December 20, 2022).

"Using the Single Camera Calibrator App - MATLAB & Simulink." 2023. https://www.mathworks.com/help/vision/ug/using-the-single-camera-calibrator-app.html (March 27, 2023).

Verstockt, Steven et al. 2011. "FireCube : A Multi-View

Localization Framework for 3D Fire Analysis." *Fire Safety Journal* 46(5): 262–75. http://dx.doi.org/10.1016/j.firesaf.2011.03.001.

Wahyono et al. 2022. "Real-Time Forest Fire Detection Framework Based on Artificial Intelligence Using Color Probability Model and Motion Feature Analysis." *Fire* 5(1): 23.

Wan, Yi, and Qisong Xie. 2016. "A Novel Framework for Optimal RGB to Grayscale Image Conversion." In *2016 8th International Conference on Intelligent Human-Machine Systems and Cybernetics, IHMSC 2016*, , 345–48.

Yang, Xiaoyuan, Yan Shi, Liuhe Chen, and Zongfeng Quan. 2010. "The Lifting Scheme for Wavelet Bi-Frames: Theory, Structure, and Algorithm." *IEEE Transactions on Image Processing* 19(3): 612–24.

Yousefi, Jamileh. 2011. "Image Binarization Using Otsu Thresholding Algorithm." *Ontario, Canada: University of Guelph* (May).

Yu, Chunyu, Zhibin Mei, and Xi Zhang. 2013a. "A Real-Time Video Fire Flame and Smoke Detection Algorithm." *Procedia Engineering* 62: 891–98. http://dx.doi.org/10.1016/j.proeng.2013.08.140.

———. 2013b. "A Real-Time Video Fire Flame and Smoke Detection Algorithm." *Procedia Engineering* 62: 891–98.

http://dx.doi.org/10.1016/j.proeng.2013.08.140.

Zhang, Dengsheng, and Guojun Lu. 2001. "Segmentation of Moving Objects in Image Sequence: A Review." *Circuits, Systems, and Signal Processing* 20(2): 143–83.

Zhang, Zhengyou. 2000. "A Flexible New Technique for Camera Calibration." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22(11): 1330–34.

———. 2021. "Camera Parameters (Intrinsic, Extrinsic)." In *Computer Vision,* Springer International Publishing, 135–40.

الخلاصة

أدى ارتفاع درجات حرارة المناخ إلى زيادة الحرائق في المباني والمزارع وحقول القمح. مستشعر الدخان القائم على الكشف عن الحرائق غير مناسب للمباني المفتوحة والكبيرة ، والمناطق الخارجية. تقترح هذه الرسالة نظامًا لاكتشاف الحرائق وتحديد موقعها استنادًا إلى كاميرا فيديو مناسبة للمناطق الداخلية والخارجية التي يمكنها تحمل التغيرات البيئية مثل الإضاءة والارتباط اللوني بألسنة اللهب والدخان.

يجمع الكشف عن الحرائق المقترح بين نهجين من أجل الحصول على كشف دقيق. يعتمد الأسلوب الأول على الكشف عن الألوان والثاني يعتمد على كشف الحركة سواء لهب النار أو الدخان. لكلا الطريقتين ، يتم التقاط فيديو إدخال RGB من الكاميرا ويتحلل باستخدام مستوى واحد من رفع تحويل الموجة (LWT) لتقليل البيانات المعالجة إلى 75٪ من حجم الإدخال دون فقدان ميزات النار. بعد ذلك ، يتم إدخال الإطارات المتحللة في خوارزميات الكشف عن اللون والحركة لتحقيق ميزات حريق محددة. بعد ذلك ، تتم معالجة إطارات الفيديو الناتجة بالعمليات المورفولجية لإزالة الكائنات غير المرغوب فيها. أخيرًا ، يتم حساب مساحة الحريق المكتشف وتحديدها بمربع إذا كانت ضمن شروط حد عتبة الحريق. تستخدم مناهج النظام المقترحة ثلاثة أنواع من حد العتبات مثل حد العتبة المتعددة الثابتة ، وحد العتبة غير التكيفية لمساحة الألوان المختلفة ، وحد العتبة التكيفية باستخدام خوارزمية تحسين سرب الجسيمات (PSO) وخوارزمية تحسين الغابة (FOA) المعتمدين على دالتي الهدف اوتسو(Otsu) وكابور(Kapur).

تم تحقيق تحديد موقع النار من خلال معايرة معلمات الكاميرا العكسية وموضع النار بالبكسل. يستخدم الأسلوب مصفوفة التحويل الإسقاطي التي يتم تعيينها بين بكسلات الإطار ومواضع العالم الحقيقي. في البداية ، يتم تدريب الخوارزمية على العديد من صور رقعة الشطرنج لتحقيق معلمات المعايرة التي تتناسب مع المواضع الحقيقية.

تستخدم MATLAB R2021b لتنفيذ نظام الكشف عن الحرائق المقترح. يتم تحقيق النتائج التجريبية للنظام المقترح للفيديوات المسجلة. تستخدم الاختبارات للفديوات المسجلة مجموعات البيانات ، بما في ذلك KUM و VisiFire و FireSense ، بالإضافة إلى مقاطع الفيديو المسجلة الخاصة بنا. تتضمن مجموعات البيانات التي تم تنزيلها 92 مقطع فيديو نار ، يعتمد على الدخان واللهب. حقق الاختبار في وضع عدم الاتصال حوالي 92.1٪ متوسط معدل الكشف الصحيح عن الحريق لنظام

اكتشاف الدخان متعدد العتبات الثابت. في المقابل ، حققت العتبة غير التكيفية القائمة على نظام الكشف عن اللهب حوالي 94٪. علاوة على ذلك ، حقق نظام كشف اللهب التكيفي حوالي 96٪.

يتم اختبار الكشف عن الحرائق في الوقت الفعلي في الأماكن الداخلية والخارجية. كان متوسط الدقة في الأماكن المغلقة 90 ٪ لنظام الكشف عن الدخان متعدد العتبات الثابت و 97.5 ٪ لنظام اكتشاف اللهب التكيفي. كان متوسط دقة الكشف عن الحرائق الخارجية لنظام الكشف عن الدخان الثابت متعدد العتبات ، وأنظمة اكتشاف اللهب غير التكيفية والتكيفية 94.1٪ و 94.4٪ و 92.4٪ على التوالي. تم تحقيق خطأ تحديد موقع الحريق عند أقل من 0.42 متر.

يمكن استخدام النظام المقترح بنجاح لاكتشاف الحرائق في الوقت الفعلي وبدقة عالية ، سواء في الداخل أو الخارج ، ولظروف بيئية مختلفة.

جمهورية العراق

وزارة التعليم العالي و البحث العلمي

جامعة كربلاء

كلية الهندسة

قسم الهندسة الكهربائية والالكترونية

نظام آلي لكشف وتحديد موقع الحريق بالإعتماد على تقنيات معالجة الصور

رسالة مقدمة الى مجلس كلية الهندسة / جامعة كربلاء وهي جزء من متطلبات نيل درجة الماجستير في علوم الهندسة الكهربائية و الالكترونية

المؤلف:

زهراء شهاب أحمد علي أصغر

باشراف :

أ.د / حيدر اسماعيل شهادي

أ.د / حوراء حسن عباس

ذو القعدة - 1444

حزيران - 2023