A

Mini Project On

**A FOREST FIRE IDENTIFICATION METHOD FOR UNMANNED AERIAL VEHICLE MONITORING VIDEO IMAGES**

(Submitted in partial fulfillment of the requirements for the award of Degree) BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

By

P.Rahul Yadav (217R1A0542)

S.Sai Charan (217R1A0549) T. Saketh (217R1A0561)

Under the Guidance of

**Dr. NUTHANAKANTI BHASKAR**

(Associate Professor)

## DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

**CMR TECHNICAL CAMPUS**

**UGC AUTONOMOUS**

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved by AICTE, New Delhi) Recognized Under Section 2(f) & 12(B) of the UGCAct.1956,

Kandlakoya (V), Medchal Road, Hyderabad-501401.

**2021-25**

## DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



**CERTIFICATE**

This is to certify that the project entitled “**A FOREST FIRE IDENTIFICATION METHOD FOR UNMANNED AERIAL VEHICLE MONITORING VIDEO IMAGES**” being submitted by **P. RAHUL YADAV (217R1A0542) , S. SAICHARAN (217R1A0549) & T. SAKETH (217R1A0561)** in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering to the Jawaharlal Nehru Technological University Hyderabad, is a record of bonafide work carried out by him/her under our guidance and supervision during the year 2024-25.

The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

#### Dr. Nuthanakanti Bhaskar Dr. A. Raji Reddy

#### Associate Professor DIRECTOR

**INTERNAL GUIDE**

**Dr. Nuthanakanti Bhaskar EXTERNAL EXAMINER**

**HoD**

**Submitted for viva voice Examination held on**

**ACKNOWLEGDEMENT**

Apart from the efforts of us, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project. We take this opportunity to express my profound gratitude and deep regard to my guide

**Dr. Nuthanakanti Bhaskar,** Associate Professor for his exemplary guidance, monitoring and constant encouragement throughout the project work. The blessing, help and guidance given by him shall carry us a long way in the journey of life on which we are about to embark. We also take this opportunity to express a deep sense of gratitude to Project Review Committee (PRC) Coordinators: **Dr. K.Maheswari, Dr. S.Suma ,Mr. J.Narasimha Rao, Mrs. K.Shilpa, Mr. K.Ranjith Reddy** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to **Dr. Nuthanakanti Bhaskar,** Head, Department of Computer Science and Engineering for providing encouragement and support for completing this project successfully.

We are obliged to **Dr. A. Raji Reddy,** Director for being cooperative throughout the course of this project. We would like to express our sincere gratitude to Sri. **Ch. Gopal Reddy,** Chairman for providing excellent infrastructure and a nice atmosphere throughout the course of this project.

The guidance and support received from all the members of **CMR Technical Campus** who contributed to the completion of the project. We are grateful for their constant support and help.

Finally, we would like to take this opportunity to thank our family for their constant encouragement, without which this assignment would not be completed. We sincerely acknowledge and thank all those who gave support directly and indirectly in the completion of this project.

#### P. RAHUL YADAV (217R1A0542) S.SAI CHARAN (217R1A0549) T.SAKEH (217R1A0561)

**ABSTRACT**

An automatic forest fire monitoring system based on UAV (unmanned aerial vehicle)-acquired video images was studied in this project. This novel method addresses current problems in forest fire monitoring practices, such as poor real-time performance and low efficiency. It aims to enable dynamic monitoring of forest fires in wild environments. This project proposes a forest fire monitoring method based on active analysis of UAV-acquired video image features to automatically detect and identify the occurrence of forest fires. The motion detection method, based on dense optical flow and background modeling, was used to extract the motion regions for eliminating background interference. Wavelet energy features and texture features were used in video images acquired by multi-rotor UAVs for forest fire monitoring. Nine sample images (eight for experiments and one for contrast) were selected. The mean values and standard deviations of the gray-level co-occurrence matrix eigenvalues (angular second moment, entropy moment, and reciprocal differential moment) were calculated to identify forest fires. The experimental results showed that the proposed algorithm effectively identifies forest fires, providing a theoretical guarantee for forest resource protection.

## LIST OF FIGURES

|  |  |  |
| --- | --- | --- |
| **FIGURE NO.** | **FIGURE NAME** | **PAGE NO.** |
| Figure 3.1 | Project Architecture | 12 |
| Figure 3.2 | Use case diagram | 14 |
| Figure 3.3 | Class diagram | 15 |
| Figure 3.4 | Sequence diagram | 17 |
| Figure 3.5 | Activity diagram | 18 |

**RESULTS AND DISCUSSIONS**

|  |  |  |
| --- | --- | --- |
| **FIGURE NO.** | **FIGURE NAME** | **PAGE NO.** |
| Figure 5.1 | Home Page | 26 |
| Figure 5.2 | Upload Video | 27 |
| Figure 5.3 | Initial Result | 28 |
| Figure 5.4 | Processed Result | 29 |

**TABLE OF CONTENTS**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ABSTRACT** | | | |  | | | |  | | | i |
| **LIST OF FIGURES** | | | | | | | |  | | | ii |
| **RESULTS AND DISCUSSIONS** | | | | | | | | | | | iii |
| **1.** | | **INTRODUCTION** | | | | | | | | | 1 |
|  | | 1.1 | PROJECT SCOPE | | | | | | | | 6 |
|  | | 1.2 | PROJECT PURPOSE | | | | | | | | 6 |
|  | | 1.3 | PROJECT FEATURES | | | | | | | | 6 |
| **2.** | | **SYSTEM ANALYSIS** | | | | | | | | | 7 |
|  | | 2.1 | PROBLEM DEFINITION | | | | | | | | 8 |
|  | | 2.2 | EXISTING SYSTEM | | | | | | | | 8 |
|  | |  | 2.2.1 | | | | LIMITATIONS OF THE EXISTING SYSTEM | | | | 9 |
|  | | 2.3 | PROPOSED SYSTEM | | | | | | | | 9 |
|  | |  | 2.3.1 | | | | ADVANTAGES OF PROPOSED SYSTEM | | | | 9 |
|  | | 2.4 | FEASIBILITY STUDY | | | | | | | | 9 |
|  | |  | 2.4.1 | | | | ECONOMIC FESIBILITY | | | | 9 |
|  | |  | 2.4.2 | | | | TECHNICAL FEASIBILITY | | | | 10 |
|  | |  | 2.4.3 | | | | SOCIAL FEASIBILITY | | | | 10 |
|  | | 2.5 | HARDWARE & SOFTWARE REQUIREMENTS | | | | | | | | 10 |
|  | |  | 2.5.1 | | | | HARDWARE REQUIREMENTS | | | | 10 |
|  | |  | 2.5.2 | | | | SOFTWARE REQUIREMENTS | | | | 10 |
| **3.** | | **ARCHITECTURE** | | | | | | | | | 11 |
|  | | 3.1 | PROJECT ARCHITECTURE | | | | | | | | 12 |
|  | | 3.2 | USECASE DIAGRAM | | | | | | | | 13 |
|  | | 3.3 | CLASS DIAGRAM | | | | | | | | 15 |
|  | | 3.4 | SEQUENCE DIAGRAM | | | | | | | | 16 |
|  | | 3.5 | ACTIVITY DIAGRAM | | | | | | | | 18 |
| **4.** | | **IMPLEMENTATION** | | | | | | | | | 20 |
|  | | 4.1 | SAMPLE CODE | | | | | | | | 21 |
| **5.** | | **RESULTS AND DISCUSSIONS** | | | | | | | | | 25 |
| **6.** | | **TESTING** | | | | | | | |  | 31 |
|  | | 6.1 | INTRODUCTION TO TESTING | | | | | | | | 32 |
|  | | 6.2 | TYPES OF TESTING | | | | | | | | 32 |
|  | |  | 6.2.1 | | | UNIT TESTING | | | | | 32 |
|  | |  | 6.2.2 | | | INTEGRATION TESTING | | | | | 32 |
|  | |  | 6.2.3 | | | FUNCTIONAL TESTING | | | | | 33 |
|  | | 6.3 | TEST CASES | | | | | | | | 33 |
|  | |  | 6.3.1 | | | UPLOADING IMAGES | | | | | 33 |
|  | |  | 6.3.2 | | | CLASSIFICATION | | | | | 34 |
| **7.** | | **CONCLUSION & FUTURE SCOPE** | | | | | | | | | 36 |
|  | | 7.1 | PROJECT CONCLUSION | | | | | | | | 37 |
|  | | 7.2 | FUTURE SCOPE | | | | | | | | 37 |
| **8.** | | **BIBLIOGRAPHY** | | | | | | |  | | 38 |
|  | 8.1 | | | | REFERENCES | | | | | | 39 |
|  | 8.2 | | | | WEBSITE | | | | | | 39 |

1. **INTRODUCTION**
2. **INTRODUCTION**

Forests play a vital role in maintaining the balance of the terrestrial ecosystem, providing numerous resources that are crucial for human survival and social development. They are not only an essential source of oxygen and biodiversity but also a major contributor to the fight against climate change by sequestering carbon dioxide. However, one of the most significant threats to forest sustainability and preservation comes in the form of forest fires, which are among the three major forest disasters globally. These fires destroy vast expanses of forest cover, release immense amounts of carbon dioxide, and cause economic and ecological devastation.

According to statistical reports, China alone experiences more than 10,000 forest fires annually, with approximately 1 million hectares burned each year—roughly 8% of the country’s total forest area. The scale of this destruction underlines the urgency for improved forest fire detection and prevention systems. Early and accurate detection is paramount in preventing the large-scale spread of forest fires, thereby mitigating their impact on both human and ecological systems.

**The Importance of Effective Fire Detection Systems**

As the severity and frequency of forest fires increase due to climate change and human activities, the need for scientific and effective detection methods becomes ever more crucial. Timely detection enables rapid response efforts, which can limit the extent of fire damage. However, detecting forest fires in their early stages remains a significant challenge due to various environmental and logistical factors, such as dense forest cover, difficult terrain, and vast areas that need to be monitored.

Existing forest fire monitoring systems include satellite monitoring, sensor networks, and video-based systems. Each of these methods comes with specific strengths and limitations:

**Satellite Monitoring**

Satellite systems provide a broad, bird’s-eye view of large geographical areas, making them ideal for monitoring remote or expansive forest regions. However, due to the inherent low refresh rates of satellite data, this method is not suitable for real-time fire detection. Satellite images are often updated only every few hours, which can delay detection and the initiation of fire-fighting efforts.

**Sensor Networks**

Ground-based sensor networks, utilizing temperature and smoke sensors, have been employed for fire detection. These networks are able to detect fire characteristics in real-time but require extensive infrastructure, including hundreds or even thousands of sensors. The

cost, complexity, and maintenance requirements of these systems, especially in remote forest regions, make them less practical on a large scale. Additionally, sensor networks are prone to technical failures due to environmental conditions and the difficulty of maintaining hardware in rugged terrains.

**Video-Based Monitoring**

Video surveillance systems have been increasingly deployed for forest fire detection. High-definition cameras installed at strategic locations allow real-time visual monitoring of specific forested areas. However, the fixed nature of these cameras limits their coverage, and the high installation and maintenance costs restrict their use to limited areas.

**Emergence of UAVs for Forest Fire Detection**

Unmanned Aerial Vehicles (UAVs), or drones, are rapidly becoming a promising alternative for forest fire detection and monitoring. With the advent of multi-rotor UAV technology, real-time monitoring has become more flexible, efficient, and cost-effective. UAVs offer several advantages over traditional systems:

**Mobility and Flexibility**

UAVs can cover vast and hard-to-reach areas by flying over dense forests and rough terrains. Their flexibility allows them to be deployed in various locations quickly, making them suitable for real-time fire detection and tracking.

**Low Manufacturing and Maintenance Costs**

Compared to ground-based sensors and satellite systems, UAVs are relatively affordable to produce and maintain. They can be rapidly deployed during high-risk fire seasons and stored when not in use.

**Real-Time Data Acquisition**

Equipped with high-resolution cameras and sensors, UAVs can capture real-time

Video footage, enabling early detection of smoke and fire. This allows for the rapid identification of fire outbreaks, giving firefighting teams more time to respond.

Given these advantages, UAV-based forest fire detection has garnered increasing attention from researchers worldwide. Various UAV models, including multi-rotor drones, have been developed to gather forest fire information efficiently and cost-effectively.

**The Role of Image and Video Analysis in Fire Detection**

While UAVs provide an ideal platform for collecting real-time video data, the challenge lies in accurately identifying fires from the vast amount of footage they capture. Traditional fire detection methods rely heavily on visual cues such as smoke and flame.

However, real-time fire detection through video streams requires advanced image processing techniques to extract relevant features and identify fire outbreaks amidst dynamic environmental conditions.

One of the most widely studied approaches for fire detection in videos is smoke detection. Since smoke often appears before visible flames, detecting smoke is a crucial part of early fire identification. Several techniques have been developed for smoke detection in video footage, including:

**Histogram Analysi**s

Histogram-based methods analyze color distribution in images to identify potential fire or smoke regions. By distinguishing between normal forest colors (e.g., green foliage) and fire-related hues (e.g., gray or white smoke), these methods can detect abnormal changes indicative of fire.

**Static and Dynamic Feature Analysis**

Forest fires have distinct dynamic features, such as flickering flames and swirling smoke, which can be identified using motion analysis. Static features like the texture of smoke or fire can also help in distinguishing them from non-fire objects.

**Video Image Segmentation**

Segmentation techniques divide video frames into different regions based on their color, texture, or motion. Smoke and fire regions are typically segmented out for further analysis. However, accurately segmenting dynamic and irregular shapes like smoke clouds can be challenging.

**Spatio-Temporal Texture Features**

Some researchers have explored using spatio-temporal textures—patterns that change over time and space in video frames—to detect fires. These textures capture the evolving nature of smoke and flames across multiple video frames, offering a more dynamic detection approach.

**Limitations of Current Methods and the Need for Improved Solutions**

Despite the progress made in fire detection technologies, many existing methods are limited to static environments and fixed monitoring ranges. This lack of dynamism makes it difficult to apply these techniques in real-world forest fire monitoring scenarios where environmental conditions are constantly changing.

For instance, most smoke detection methods assume that the camera’s viewpoint is fixed, making it difficult to track smoke across different video frames. Moreover, many of these techniques have been developed for urban or indoor fire detection, where lighting and

environmental conditions are controlled. In contrast, forest fire detection must contend with natural variability, such as shifting weather patterns, dense tree cover, and uneven terrain.

In addition, the vast distances covered by forests require continuous monitoring, which many traditional systems are not capable of providing. While UAVs offer a mobile and flexible platform, the challenge remains to develop image processing algorithms capable of detecting fires in dynamic, moving environments.

**Contributions of This Study**

To address these challenges, the present study proposes a novel forest fire identification method using UAV video analysis. This method incorporates active image analysis techniques to improve the accuracy of forest fire detection while considering the dynamic nature of UAV-based video footage. Key contributions include:

**Real-Time Detection**

By leveraging the mobility of UAVs and advanced image processing techniques, this method allows for continuous, real-time monitoring of forested areas, enabling early detection and intervention.

**Improved Background Discontinuity Handling**

The method addresses the issue of background discontinuity, a common problem in UAV-based video monitoring where the background changes rapidly due to UAV movement. This improvement leads to more accurate fire detection, even in complex and changing environments.

**Multi-Frame Analysis**

Unlike traditional methods that analyze individual frames, the proposed method uses multi-frame analysis to track fire-related features such as smoke and flames across consecutive video frames. This approach helps in distinguishing transient environmental changes from actual fire incidents.

By enhancing fire detection accuracy and overcoming the limitations of previous approaches, this method offers a promising solution for effective forest fire monitoring using UAVs.

### PROJECT SCOPE

##### This project is titled “A Forest Fire Identification Method for Unmanned Aerial Vehicle Monitoring Video Images”. The software system aims to facilitate the automatic detection and identification of forest fires by analyzing video images captured by UAVs (Unmanned Aerial Vehicles). It uses advanced image processing techniques and machine learning algorithms to classify smoke and fire patterns from UAV-acquired videos. The system extracts motion and texture features from the images, processes the data, and identifies forest fires in real-time. The key focus is on improving accuracy, efficiency, and the ability to monitor dynamic, wide-ranging forest environments.

### PROJECT PURPOSE

This system has been developed to improve the detection and monitoring of forest fires. By using real-time UAV-acquired video data, the system can accurately identify forest fires and reduce the response time to such incidents. Traditional methods are inefficient in terms of cost and real-time data, but this system overcomes those challenges by providing continuous monitoring, even in dynamic conditions. The system also ensures reliable and automatic identification of forest fires, which helps in safeguarding valuable forest resources and preventing large-scale environmental damage.

### PROJECT FEATURES

The main features of this project include the integration of UAV technology and machine learning algorithms for forest fire detection. The system serves as a solution to current fire monitoring challenges by offering real-time image analysis and fire identification. Multiple motion and texture analysis techniques are used to enhance the accuracy of detection. The system presents its results and proposals to users, who can then monitor the detected fires and take the necessary action. It ensures user feedback by offering a loop of verification, where users can request system adjustments based on their needs, ensuring optimal system performance.

# SYSTEM ANALYSIS

### 2.SYSTEM ANALYSIS

System analysis is a critical phase in the development of the "**Forest Fire Identification Method for Unmanned Aerial Vehicle Monitoring Video Images**". In this phase, the existing system is thoroughly studied and analyzed to gain an in-depth understanding of its functioning. The system analyst acts as an investigator, exploring the existing system to identify challenges and areas of improvement. The goal of system analysis is to determine **"what must be done to solve the problem"** by evaluating the inputs, outputs, and processes involved.

During this analysis, the system is examined in its entirety. This includes identifying data inputs such as UAV video feeds, image processing techniques, and the detection algorithms. The relationships between these components and their interactions with external systems (such as operators, UAV control, and real-time response systems) are studied. By the end of this phase, the analyst will have a clear understanding of what is required to improve forest fire monitoring using UAVs.

### PROBLEM DEFINITION

In this project, a detailed examination of the forest fire monitoring process is conducted using techniques like **video processing**, **motion detection**, **color and texture analysis**, and **feature extraction**. Data collected from UAV-acquired video images is processed to identify areas where the current system is lacking, particularly in terms of real-time performance, detection accuracy, and adaptability to dynamic environments.

### EXISTING SYSTEM

Traditional monitoring methods cannot collect forest fire video information in real-time and effectively. At present, due to the characteristics of heavy load, long duration, and strong wind resistance, the eight-rotor unmanned aerial vehicle is widely used in the forest fire monitoring field.

The eight-rotor aircraft is driven by eight independent motors, where the adjacent motors rotate in the opposite direction to eliminate the torque caused by motor rotation. The aircraft can control six degrees of freedom by adjusting the rotational speed of the eight rotors.

#### 

#### 2.2.1 LIMITATIONS OF EXISTING SYSTEM

* + - * Less Accuracy
      * Less Efficiency

### 2.3 PROPOSED SYSTEM

### Traditional video-based forest fire monitoring equipment usually consists of fixed cameras deployed on mountain tops, capturing static backgrounds, which are only suitable for long-distance and large-field monitoring. These methods are unable to handle other conditions, such as foggy video captured during the morning. This paper proposes a novel forest fire detection method based on active image analysis to address these limitations.

### 2.3.1 ADVANTAGES OF THE PROPOSED SYSTEM

* High Accuracy
* High Efficiency

### 2.4 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase, and a business proposal is put forth with a general plan for the project and some cost estimates. During system analysis, the feasibility study of the proposed system is carried out to ensure that the proposed system is not a burden to the company. For feasibility analysis, understanding the major requirements of the system is essential.

Three key considerations involved in the feasibility analysis are:

* ECONOMIC FEASIBILITY
* TECHNICAL FEASIBILITY
* SOCIAL FEASIBILITY

### 2.4.1 ECONOMIC FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of funds the company can allocate for the research and development of the system is limited, and expenditures must be justified. The system must be developed within the budget, and this was achieved because most of the technologies used are freely available, with only customized products needing to be purchased.

### 

### 2.4.2 TECHNICAL FEASIBILITY

This study is conducted to check the technical feasibility, i.e., the technical requirements of the system. Any system developed must not place high demands on the available technical resources, as this would lead to increased pressure on the client. The developed system must have modest requirements, with minimal or no changes needed for implementation.

### 2.4.3 SOCIAL FEASIBILITY

This aspect of the study checks the level of acceptance of the system by the users. It includes the process of training users to efficiently use the system. Users must not feel threatened by the system; instead, they should accept it as a necessity. The level of acceptance by the users depends on the methods employed to educate them about the system and familiarize them with it. Their confidence must be raised so they can provide constructive criticism, which is welcomed since they are the final users of the system.

### 2.5 HARDWARE & SOFTWARE REQUIREMENTS

**2.5.1 HARDWARE REQUIREMENTS :**

Hardware interfaces specifies the logical characteristics of each interface between the software product and the hardware components of the system. The following are some hardware requirements.

|  |  |  |
| --- | --- | --- |
| * Processor | : | Intel Core i5 |
| * Hard disk | : | 40GB and Above. |
| * RAM | : | At least 8GB |
| * Monitor | : | 5 inches or above. |

### 2.5.2 SOFTWARE REQUIREMENTS :

Software Requirements specifies the logical characteristics of each interface and software components of the system. The following are some software requirements,

* Operating system : Windows 11
* Languages : Python(v3.7)
* Framework : OpenCV, numpy, Tkinter
* Designing : Tkinter+

# 3. ARCHITECTURE

## 

## 3.ARCHITECTURE

### 3.1 PROJECT ARCITECTURE

This project architecture shows the procedure followed for Forest Fire detection machine learning, starting from input to final prediction.

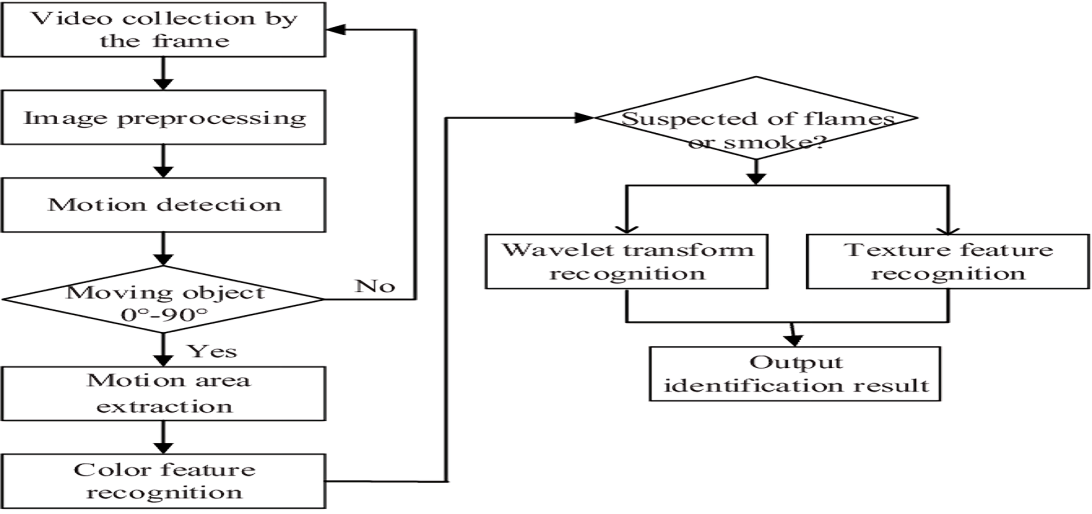


Figure 3.1: Project Architecture to Identify Forest Fire Using UAV

### The above figure 3.1 illustrates about the flow of the process regarding identification of forest fires by using various processing techniques. Below are the detailed description about ecah step.

**Video collection by the frame**: The UAV collects video footage, capturing it frame by frame, allowing for detailed and sequential analysis of the video data.

**Image preprocessing**: This step involves enhancing the video frames, such as adjusting contrast or noise reduction, to make the subsequent steps of motion and fire detection more efficient.

**Motion detection**: This identifies movement within the frames, isolating any dynamic elements in the video, which are crucial for detecting potential fire or smoke activity.

**Moving object 0°-90°**: The system analyzes the movement angle of objects within the video. If the movement falls within the 0° to 90° range, the object is flagged for further analysis.

**Motion area extraction**: After motion detection, the area where the movement is detected is extracted for focused examination, reducing processing time for irrelevant areas.

**Color feature recognition**: The extracted motion areas are analyzed for color characteristics, specifically looking for the colors typically associated with fire and smoke, such as shades of red, orange, and gray.

**Suspected of flames or smoke?**: Based on the motion and color recognition, the system determines whether the detected object is potentially smoke or fire, triggering the next step for detailed analysis.

**Wavelet transform recognition**: This method breaks down the image into different frequency components to identify detailed features of flames or smoke, improving detection accuracy.

**Texture feature recognition**: The system evaluates the texture characteristics of the identified objects (such as the smoothness or roughness of smoke or fire), aiding in accurate identification.

**Output identification result**: After all analyses, the system produces an output identifying whether a fire or smoke is present in the frame, allowing for action or further monitoring.

### 3.2 USE CASE DIAGRAM

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

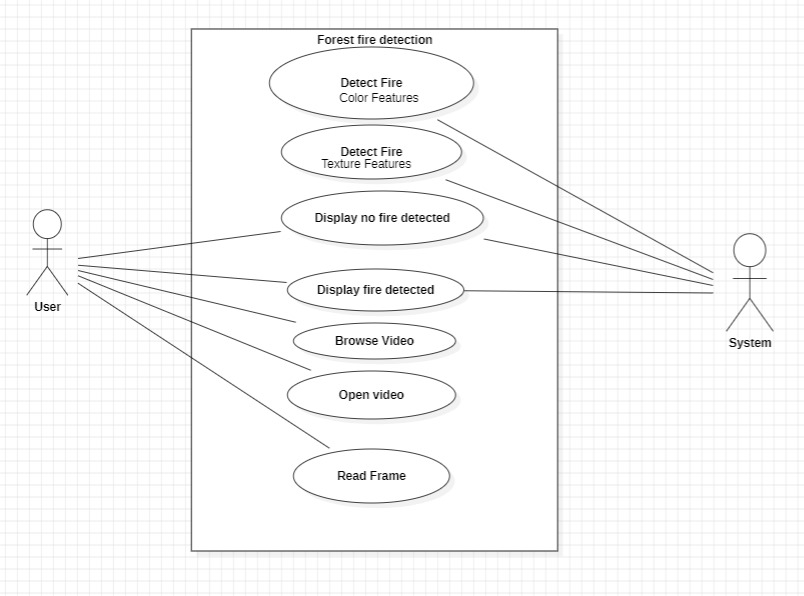


Figure 3.2: Use Case Diagram for user for Forest Fire Identification using UAV

As shown in the above Figure 3.2, this use case diagram illustrates the interaction between a **User** and a **System** designed for forest fire detection based on video data. The diagram represents the system's core functionalities and how the user interacts with the system to detect fire in video footage.

The **User** interacts with the system by first browsing and selecting a video through the **Browse Video** option. Once the video is selected, the **Open Video** use case is triggered, and the system begins reading the video, frame by frame, using the **Read Frame** process. The system processes each video frame to detect potential signs of a forest fire.

The fire detection process is divided into two methods: **Detect Fire (Color Features)** and **Detect Fire (Texture Features)**. In the first method, the system analyzes the color spectrum of the frames, searching for typical fire colors like shades of red, orange, and yellow. In the second method, the system evaluates texture patterns that are characteristic of fire and smoke. These texture features help to differentiate fire from other objects in the scene that may share similar colors but are not fire.

After the system completes its analysis, it displays the results to the user. If no fire is detected in the analyzed frames, the system triggers the **Display No Fire Detected** message, informing the user that the footage is clear of any fire threats. Conversely, if fire is detected based on either color or texture features, the system generates a **Display Fire Detected** message, alerting the user to the presence of a fire.

### 3.3 CLASS DIAGRAM

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

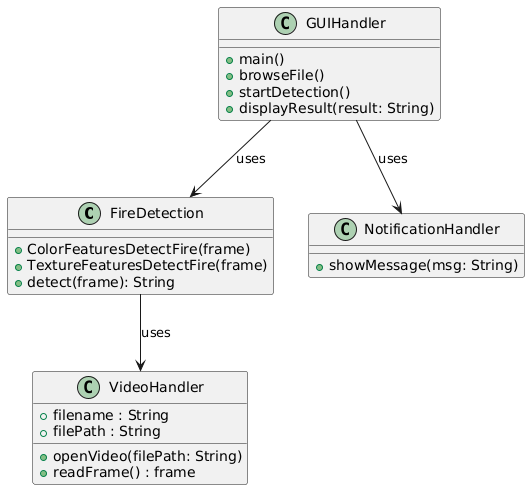


Figure 3.3: Class Diagram for Forest fire identification using UAV

The above figure 3.3 provides information about the structure of a forest fire detection system using a class diagram representation. The GUIHandler class is the user interface, allowing the user to select videos and initiate fire detection through methods like browseFile() and startDetection(). It also displays the detection results using displayResult(result: String).

The FireDetection class is responsible for analyzing video frames to detect fire. It uses two methods, ColorFeaturesDetectFire(frame) and TextureFeaturesDetectFire(frame), to detect fire based on color and texture features. The final result is returned via detect(frame): String.

The VideoHandler class manages the video file, opening it using openVideo(filePath: String) and reading frames through readFrame(). It supplies frames to the FireDetection class for analysis.

The NotificationHandler class displays messages to the user via showMessage(msg: String), such as whether a fire has been detected.

In terms of interaction, the user starts by using the GUIHandler to select a video file and begin the detection process. The GUIHandler then calls the FireDetection class, which processes the video frames with the help of the VideoHandler. After analyzing the frames, the FireDetection class returns the detection result to the GUIHandler, which uses the NotificationHandler to display the outcome to the user.

### 

### 3.4 SEQUENCE DIAGRAM

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

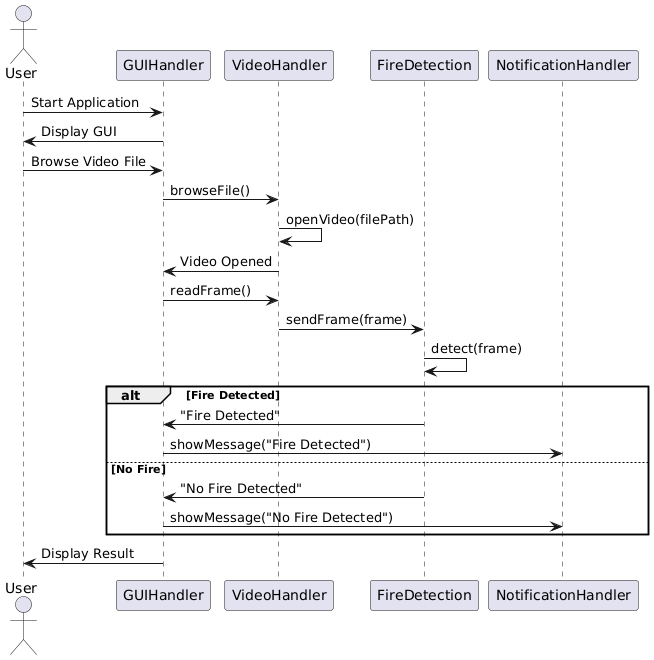


Figure 3.4: Sequence Diagram for Forest Fire identification using UAV

Figure 3.4 is a sequence diagram that illustrates the workflow of a forest fire detection system, outlining the interaction between the user and various system components. The process begins with the user starting the application, which prompts the GUIHandler to display the graphical user interface (GUI). The user then browses for and selects a video file for fire detection.

Once the user selects the file, the GUIHandler triggers the browseFile() method to communicate with the VideoHandler. The VideoHandler opens the selected video using the openVideo(filePath) method, confirming that the video file has been successfully loaded.

After loading the video, the GUIHandler instructs the VideoHandler to read individual frames using the readFrame() method. The VideoHandler extracts a frame and sends it to the FireDetection class for analysis through the sendFrame(frame) method.

At this point, the FireDetection class processes the frame using the detect(frame) method, analyzing whether fire is present. Two possible outcomes can occur: if fire is detected, the FireDetection class sends a "Fire Detected" message back, which is displayed to the user via the NotificationHandler with the showMessage("Fire Detected") method. If no fire is detected, the system sends a "No Fire Detected" message, which is similarly displayed to the user using showMessage("No Fire Detected").

Finally, the detection result is shown on the GUI through the GUIHandler, completing the process.

### 3.5 ACTIVITY DIAGRAM

It describes about flow of activity states.

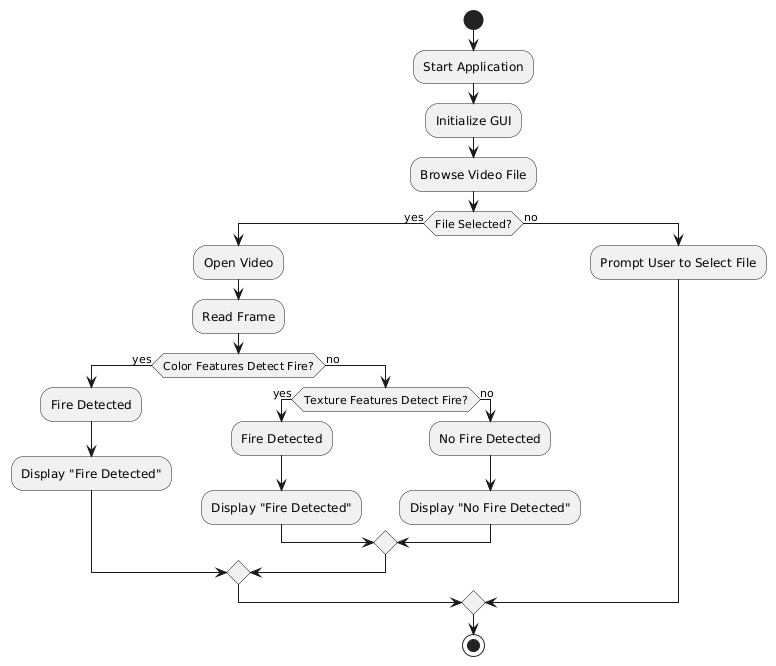


Figure 3.5: Activity Diagram for Forest Fire detection using UAV

As shown in the above Figure 3.5 ,the activity diagram describes the process of detecting forest fires using a video-based detection system. The process starts with the user launching the application, which initializes the graphical user interface (GUI). After the

application is running, the user is prompted to browse and select a video file for analysis. If the user does not select a file, the system prompts them again until a file is chosen.

Once a video file is selected, the system opens the video and begins reading the frames. The first step in detecting a fire is through the analysis of color features in the video frames. If the system detects fire based on color, it displays a "Fire Detected" message. If no fire is detected using color features, the system moves on to analyze the texture features within the video frame. If fire is detected through texture features, the system again displays a "Fire Detected" message.

If neither color nor texture features indicate the presence of fire, the system concludes that no fire is detected and displays a "No Fire Detected" message. The process continues for each frame of the video, repeating the analysis until all frames have been processed or a fire is detected. This ensures comprehensive monitoring of the video footage to accurately detect forest fires.

# 4.IMPLEMENTATION

### IMPLEMENTATION

### Algorithms and Techniques Used

1. **Color Feature Detection**:

* The fire detection process is primarily based on identifying colors typical of fire, such as shades of orange and yellow, using the HSV (Hue, Saturation, Value) color model.
* A range of HSV values is set for fire-like colors, which helps create a mask to detect areas in the video frame that match this color profile.

1. **Motion Detection**:

* The code uses frame differencing, which detects changes between consecutive video frames to identify motion. This helps distinguish static areas from those that exhibit movement (indicating fire).
* For each frame, it converts to grayscale, blurs it to reduce noise, and computes the absolute difference between the current and previous frame. This helps in identifying motion in the video.

**PROCEDURE**

1. **GUI Setup:**

* The Tkinter library creates a GUI window with buttons and text areas. The main window is titled with the project name and configured with a specific size and background color.

1. **upload() Function:**

* This function opens a file dialog to upload a video file. Once a file is selected, its path is displayed in a text area in the GUI.

1. **ColorFeaturesDetectFire() Function:**

* This function receives a video frame, blurs it, and converts it to the HSV color space.
* A color range (18 to 35 for hue) defines fire-like colors. The cv2.inRange() function isolates areas within this range, creating a mask.
* The mask highlights fire-colored regions, and cv2.countNonZero() counts non-zero pixels in the mask to quantify fire-colored pixels.
* If the count exceeds 4000, the function returns “Fire detected”; otherwise, it returns “No Fire Detected”.

1. **detectFire() Function:**

* This function reads frames from the uploaded video and passes each frame to ColorFeaturesDetectFire() for color detection.
* After color detection, it proceeds with motion detection:
* The frame is converted to grayscale and blurred to reduce noise.
* If it’s the first frame, it’s saved as the previous\_frame.
* The absolute difference between the previous\_frame and the current frame detects movement.
* The difference frame undergoes dilation (to fill small gaps) and thresholding to isolate moving regions.
* Contours in this thresholded frame indicate areas of movement.
* The result from color detection is displayed on each frame. Pressing 'q' stops the video playback.

1. **GUI Components:**

* Buttons for uploading a video and running detection functions trigger upload() and detectFire() functions.
* The text area shows the file path of the uploaded video and any messages or results.

This combination of color detection and motion detection enhances fire detection accuracy by focusing on both fire-like colors and movement, reducing false positives for static orange objects.

**4.1 SAMPLE CODE**

from tkinter import messagebox

from tkinter import \*

from tkinter import simpledialog

import tkinter

import numpy as np

from tkinter import simpledialog

from tkinter import filedialog

import os

import cv2

main = tkinter.Tk()

main.title("A Forest Fire Identification Method for Unmanned Aerial Vehicle Monitoring Video Images") #designing main screen

main.geometry("1300x1200")

global filename

def ColorFeaturesDetectFire(frame):

  msg = "No Fire Detected"

    blur = cv2.GaussianBlur(frame, (21, 21), 0)

    hsv = cv2.cvtColor(blur, cv2.COLOR\_BGR2HSV)

    lower = [18, 50, 50]

    upper = [35, 255, 255]

    lower = np.array(lower, dtype="uint8")

    upper = np.array(upper, dtype="uint8")

    mask = cv2.inRange(hsv, lower, upper)

    output = cv2.bitwise\_and(frame, hsv, mask=mask)

    no\_red = cv2.countNonZero(mask)

    print(no\_red)

    if int(no\_red) > 4000:

        msg = "Fire detected"

    return msg, mask

def upload():

    global filename

    filename = filedialog.askopenfilename(initialdir="UAV\_Videos")

    text.delete('1.0', END)

    text.insert(END,filename+" loaded\n");

def detectFire():

    global filename

    video = cv2.VideoCapture(filename)

    previous\_frame = None

while(True):

        ret, frame = video.read()

        if ret == True:

            msg, temp = ColorFeaturesDetectFire(frame)

img\_rgb = cv2.cvtColor(src=frame, code=cv2.COLOR\_BGR2RGB)

          prepared\_frame = cv2.cvtColor(img\_rgb, cv2.COLOR\_BGR2GRAY)

            prepared\_frame = cv2.GaussianBlur(src=prepared\_frame, ksize=(5, 5), sigmaX=0)

            if (previous\_frame is None):

                previous\_frame = prepared\_frame

                continue

            if (previous\_frame is None):

                previous\_frame = prepared\_frame

                continue

  diff\_frame=cv2.absdiff(src1=previous\_frame, src2=prepared\_frame)

            previous\_frame = prepared\_frame

            kernel= np.ones((5, 5))

            diff\_frame = cv2.dilate(diff\_frame, kernel, 1)

            thresh\_frame=cv2.threshold(src=diff\_frame, thresh=20, maxval=255, type=cv2.THRESH\_BINARY)[1]

            contours,\_=cv2.findContours(image=thresh\_frame, mode=cv2.RETR\_EXTERNAL,method=cv2.CHAIN\_APPROX\_SIMPLE)#cv2.drawContours(image=frame, contours=contours, contourIdx=-1, color=(0, 255, 0), thickness=2, lineType=cv2.LINE\_AA)

            '''

         for contour in contours:

          if cv2.contourArea(contour) < 50:

                    continue

               (x, y, w, h) = cv2.boundingRect(contour)

               cv2.rectangle(img=img\_rgb, pt1=(x, y), pt2=(x + w, y + h), color=(0, 255, 0), thickness=2)

            '''

        cv2.putText(frame, msg, (10, 50), cv2.FONT\_HERSHEY\_SIMPLEX, 0.7, (255, 0, 0), 2)

            cv2.imshow('Fire Detector', frame)

            cv2.imshow("Motion Image",temp)

            if cv2.waitKey(600) & 0xFF == ord('q'):

                break

    cv2.destroyAllWindows()

font = ('times', 16, 'bold')

title = Label(main, text='A Forest Fire Identification Method for Unmanned Aerial Vehicle Monitoring Video Images')

title.config(bg='LightGoldenrod1', fg='medium orchid')

title.config(font=font)

title.config(height=3, width=120)

title.place(x=0,y=5)

font1 = ('times', 12, 'bold')

text=Text(main,height=25,width=140)

scroll=Scrollbar(text)

text.configure(yscrollcommand=scroll.set)

text.place(x=10,y=200)

text.config(font=font1)

font1 = ('times', 12, 'bold')

uploadButton = Button(main, text="Upload UAV Forest Fire Video", command=upload)

uploadButton.place(x=50,y=100)

uploadButton.config(font=font1)

preButton = Button(main, text="Run Motion Detection, Colour Features and Wavlet Transfrom to Detect File", command=detectFire)

preButton.place(x=350,y=100)

preButton.config(font=font1)

main.config(bg='OliveDrab2')

main.mainloop()

# RESULTS AND DISCUSSIONS

**5.RESULTS AND DISCUSSIONS**

### 5.1 UPLOAD RESULT

### In this Screenshot we can see the Home page for forest detection application



Figure 5.1: Home Page of the application

The above screenshot(Figure 5.1) shows the graphical user interface (GUI) of a forest fire detection application designed for UAV (unmanned aerial vehicle) monitoring videos. The title at the top clearly indicates the purpose of the tool: **"A Forest Fire Identification Method for Unmanned Aerial Vehicle Monitoring Video Images."** The interface features two primary buttons.

The first button, **"Upload UAV Forest Fire Video,"** allows the user to upload video footage captured by UAVs for analysis. After the video is uploaded, the user can click the second button, **"Run Motion Detection, Colour Features, and Wavelet Transform to Detect Fire,"** which triggers the fire detection process. This process leverages motion

detection, color feature analysis, and wavelet transforms to identify signs of fire in the video frames.

**5.2 UPLOAD VIDEO**

In this Screenshot we can see the Video to get uploaded for forest fire detection

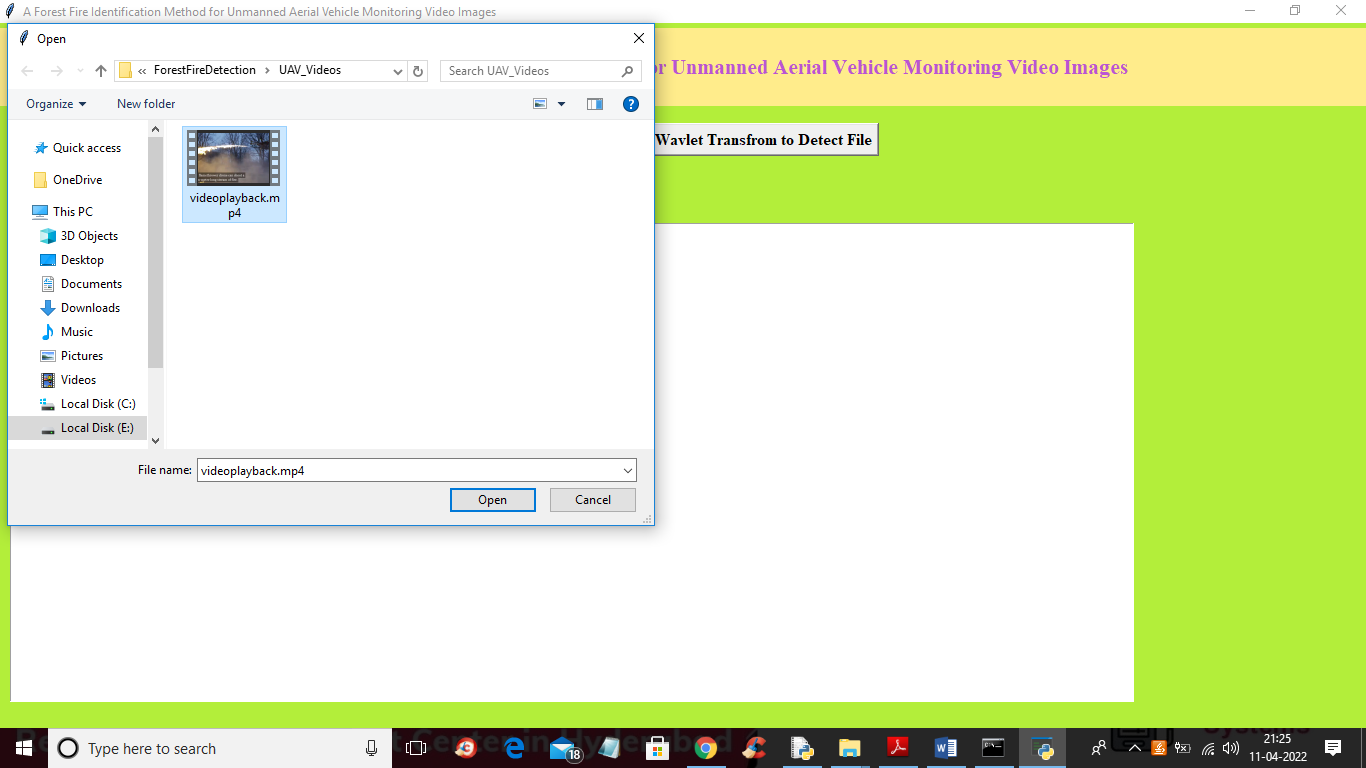


Figure 5.2 : Upload video

The figure 5.2 demonstrates the file upload process, which is triggered when the user presses the **"Upload UAV Forest Fire Video"** button. A dialog box has opened, allowing the user to browse their local computer for the video file they wish to upload for fire detection.

In this case, the user is navigating to a folder named **"UAV\_Videos"** within a directory labeled **"ForestFireDetection"**. The video is selected from the file named **"videoplayback.mp4"**. This file is likely UAV footage of a forest, which will be processed by the system once uploaded.

### 

### 5.3 INITIAL RESULT

### In this Screenshot we can see the intial result of the video where fire is not detected

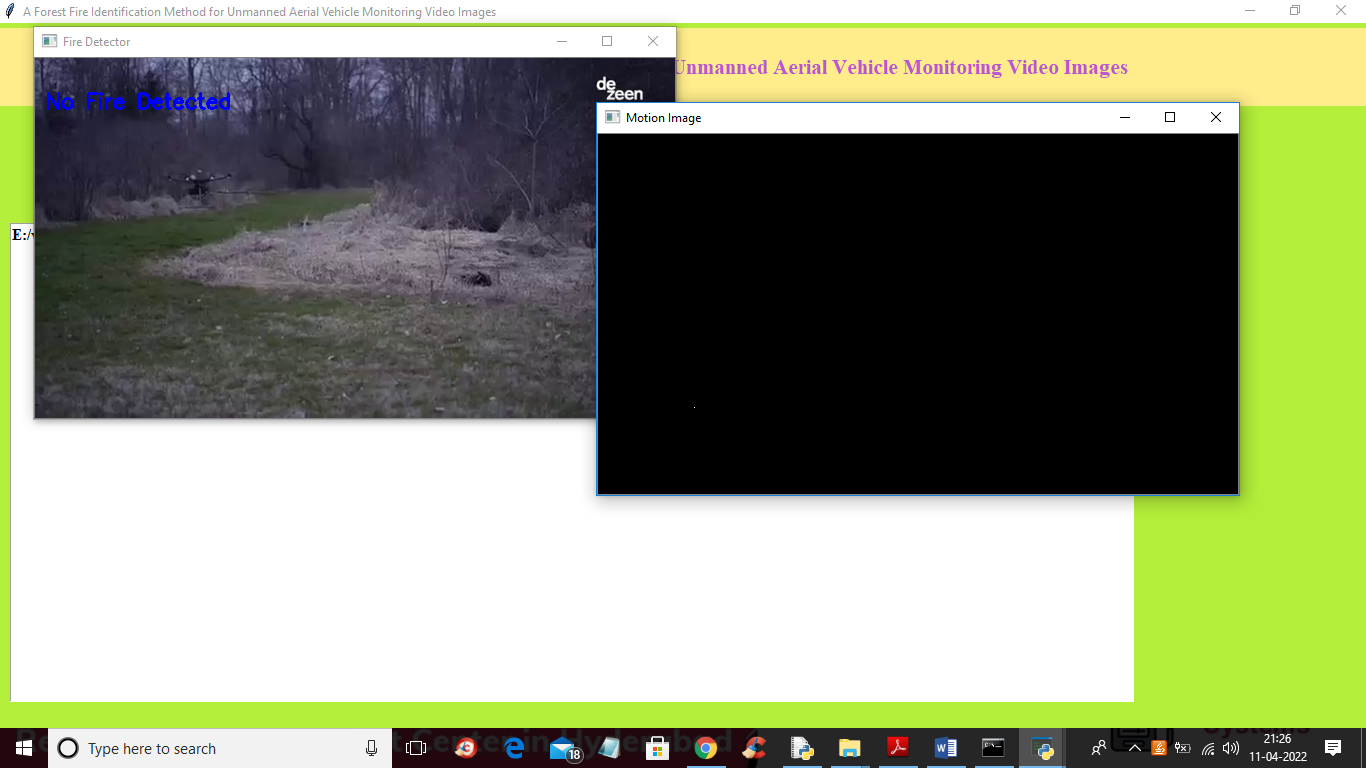


Figure 5.3: Initial result of forest fire detection using UAV(no fire detected)

The above screenshot(Figure 5.3) demonstrates the result of running the forest fire detection system after uploading a UAV-captured video. The system has processed the video, and two distinct windows are visible.

### Left Window – Fire Detector Output:

* The left window displays the actual UAV video footage, which shows a forested area. The landscape appears calm, with no visible signs of fire.
* In the top left corner of the window, a message reads **"No Fire Detected"** in blue text, indicating that the system has analyzed this frame and found no evidence of a fire in the video.
* The video appears to be playing smoothly, and the UAV is monitoring a portion of the forest, likely for early detection of any potential fires.

**Right Window – Motion Image:**

* The smaller window on the right, labeled **"Motion Image"**, appears to show a black screen. This window likely represents the output from the motion detection algorithm used by the system.
* Since there is no visible motion in this frame (and the system detected no fire), the motion image remains entirely black, indicating no significant activity or dynamic changes detected in this portion of the video.

### 5.4 PROCESSED RESULT

In this screenshot we can see the result of fire detection.

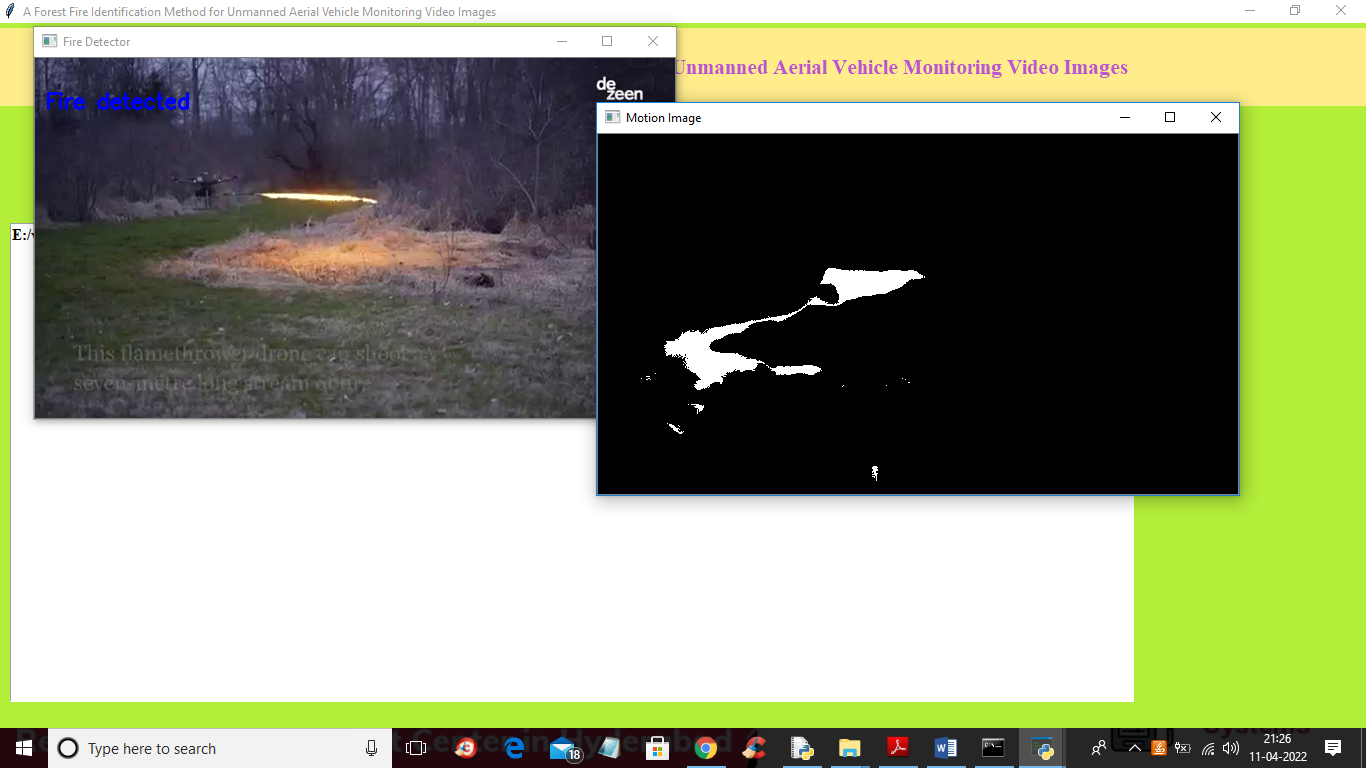


Figure 5.4: Processed result(Fire detected)

The screenshot( Figure 5.4) shows a result for our input. In this result, two windows are visible, each providing different aspects of the fire detection process.

On the left side, the first window titled "Fire Detector" displays an image of an outdoor scene, where a UAV (Unmanned Aerial Vehicle) is hovering over an area, and fire or flames are visible on the ground. In the upper-left corner of the window, the system has detected the fire, and the text "Fire detected" is shown in blue, indicating the system’s successful identification of the flames in the video footage.

On the right side, there is a second window titled "Motion Image," displaying a black-and-white binary image. This image represents the system’s motion detection algorithm, where the white regions correspond to areas with significant motion or changes detected, likely correlating with the flames in the scene. This binary motion image helps the system isolate areas of interest, such as the fire, from the rest of the scene.

# TESTING

## TESTING

### INTRODUCTION TO TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discoverevery conceivable fault or weakness in a work product. It provides a way to check the functionality of components, subassemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

### 6.2 TYPES OF TESTING

**6.2.1 UNIT TESTING**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

### 6.2.2 INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

### 6.2.3 FUNCTIONAL TESTING

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

|  |  |
| --- | --- |
| Valid Input | identified classes of valid input must be accepted. |
| Invalid Input | identified classes of invalid input must be rejected. |
| Functions | identified functions must be exercised. |
| Output | identified classes of application outputs must be exercised. |

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes.

### 6.3 TEST CASES

**6.3.1 UPLOADING IMAGES**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test case ID | Test case name | Purpose | Test Case | Output |
| 1 | User uploads video | Use it for identification | The user uploads a video containing fire | Uploaded successfully |
| 2 | User uploads 2nd video | Use it for identification | The user uploads the a non-fire video | Uploaded successfully |

**TABLE 6.1**

The above Table 6.1 gives the representation of the different test cases along eith its name, purpose and output. Each test case has a unique **Test Case ID** to differentiate it from others and a **Test Case Name** that provides a brief description of the test being performed. The **Purpose** column explains the goal of the test, while the **Test Case** column details the specific action that the user will perform. Finally, the **Output** column shows the expected result after executing the test.

**Test Case 1**, named "User uploads video," is designed to ensure that the system can successfully process and upload a video containing fire for identification purposes. The user performs this test by uploading a video that features fire, and the expected output is that the system uploads the video successfully, indicating readiness for further analysis.

**Test Case 2**, called "User uploads 2nd video," aims to test whether the system can handle videos without fire and still process them without issues. The user conducts this test by uploading a non-fire video. The expected output for this test is also a successful upload, confirming that the system can accept and process general video inputs without errors.

These test cases help confirm the functionality of the video upload process in the forest fire identification system, ensuring it performs as expected when handling both fire-containing and non-fire videos.

**6.3.2 CLASSIFICATION**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test case ID | Test case name | Purpose | Input | Output |
| 1 | Classification test 1 | To check if the classifier performs its task | A video is given | fire is detected |
| 2 | Classification test 2 | To check if the classifier performs its task | A non fire video is given | Fire not detected. |

**TABLE 6.2**

The above table (Table 6.2) provides the information about the classification tests done during the project execution.These tests are essential to verify the effectiveness of the system's classifier in identifying whether an input video contains fire. Each column in the table serves a specific purpose: **Test Case ID** uniquely identifies each test, **Test Case Name** provides a descriptive label for the test, **Purpose** explains the objective of the test, **Input** specifies the type of video provided to the classifier, and **Output** states the expected result.

**Test Case 1**, titled "Classification test 1," aims to confirm that the classifier can accurately detect fire when a video containing fire is provided. The input for this test is a video that shows fire, and the expected output is the detection of fire, verifying that the system can successfully classify videos with fire content.

**Test Case 2**, named "Classification test 2," is designed to check whether the classifier can correctly identify when there is no fire in the input video. For this test, a non-fire video is given as input, and the expected output is that the classifier should not detect any fire. This result ensures that the system can distinguish between videos that contain fire and those that do not.

Overall, these classification test cases validate the reliability and accuracy of the system's ability to identify fire in videos, ensuring its effectiveness for the intended purpose of forest fire detection.

**CONCLUSION**

**7. CONCLUSION & FUTURE SCOPE**

* 1. **PROJECT CONCLUSION**

In this paper, the monitoring information of forest fires is obtained by a multi-rotor unmanned aerial vehicle (UAV) equipped with video acquisition equipment. The experimental sample image is extracted frame by frame. This paper proposes a forest fire monitoring method for UAV video images based on active analysis. Real-time monitoring and automatic recognition of forest fires are achieved by analyzing static characteristics of forest fires, such as angular second moment, entropy, and reciprocal differential moment. The experimental results show that the proposed algorithm can effectively identify forest fires, achieving the real-time monitoring goals of forest fires using a multi-rotor UAV.

### FUTURE SCOPE

The future of this forest fire detection method involves combining data from various sources like satellite images, weather data, and IoT sensors for more accurate fire detection. Future advancements will refine machine learning and use deep learning techniques like CNNs for image data and RNNs for time-based data to boost detection accuracy. Improving how fire patterns are captured over time and space will help detect fires more precisely, using methods like spatial interpolation and time series analysis. Adding methods to handle uncertainty, such as Bayesian approaches and ensemble models, will make predictions more reliable. Integrating real-time monitoring with decision-making tools will provide quicker and more effective responses to forest fires for stakeholders.

# 8. BIBLIOGRAPHY

## 8. BIBLIOGRAPHY

### 8.1 REFERENCES

### [1] G. Lewis S, Clarke M. Forest plots: trying to see the wood and the trees.[J]. Bmj, 2001.

### [2] Turner D, Lewis M, Ostendorf B. Spatial indicators of fire risk in the arid and semi-arid zone of Australia[J]. Ecological Indicators, 2011.

### [3] Adab H. Using Probabilistic Methods to Evaluate Landfire Hazard[C]// International Conference on Environmental Engineering. 2016.

### [4] Lei Z, Lu J. Distributed coverage of forest fire border based on WSN[C]// International Conference on Industrial and Information Systems. IEEE, 2010.

### [5] Jadhav P., Deshmukh V., et al. Forest fire monitoring system based on Zig-Bee wireless sensor network[J]. Internation Journal of Emerging Technology and Advanced Engineering, 2012.

### [6] Liu L Y, Fan X J. The Design of System to Texture Feature Analysis Based on Gray Level Co-Occurrence Matrix[J]. Applied Mechanics & Materials, 2015.

### 8.2 WEBSITE

### [1] <https://ieeexplore.ieee.org/document/8756696>