Automatic Fire Detection and Warning System Adaptable to Any System

Zülal Karın, Ayca Tanışlı, Fuat Yiğit Koçyiğit

Computer Engineering, TED University

zulal.karin@tedu.edu.tr

zayca.tanisli@tedu.edu.tr

fyigit.kocyigit@tedu.edu.tr

Abstract— **Strong The threat of fires in our homes, institutions, and workplaces is a serious concern that requires us to develop reliable detection and alert systems. While existing systems have proven to be effective, they often require significant human intervention and can be slow to respond to emergencies. Our paper introduces an innovative automatic fire detection and warning system that can be easily integrated into any environment that uses digital images obtained from video surveillance.**

Our system, which is written in Python, leverages the power of computer vision through OpenCV to provide a highly accurate and reliable fire detection mechanism. Additionally, the system uses smtplib to instantly send email notifications to users when a fire is detected. This allows users to take immediate action and ensures that potential fire threats are addressed as quickly as possible.

Moreover, the system also uses playsound for audible alerts, which provides an additional layer of safety. This way, not only do users receive email notifications, but they are also alerted to the presence of a fire through an audible alarm.

Overall, our system provides an efficient and rapid response to potential fire threats, enhancing the safety of the premises. The system is designed to be adaptable and can be customized to suit the specific needs of any environment. With our automatic fire detection and warning system, you can have peace of mind knowing that you and your property are well-protected against the threat of fires.

Keywords— **Automatic Fire Detection, OpenCV, Python, Email Notification, Video Surveillance.**

I. Introduction

Fire safety is an issue of utmost importance, and the need for efficient and rapid response systems is more pressing than ever. While traditional fire detection systems, such as smoke detectors, can be effective, they are not always reliable when it comes to instant detection and alert. Therefore, it is imperative to develop more advanced and reliable detection systems that can promptly detect and alert users in case of a fire.

To address this need, we present an automatic fire detection and warning system that utilizes real-time video surveillance to ensure efficient and prompt detection of fires. The system is designed to trigger an alarm and send an email notification to the user as soon as a fire is detected. This ensures that the user can take immediate action and prevent the fire from spreading.

Our system represents a significant improvement over traditional fire detection systems, as it offers a more reliable and efficient approach to fire safety. This paper details the implementation and operation of the system, providing valuable insights into the development of advanced detection systems for fire safety. With our system, users can rest assured that they are protected from the dangers of fire, and can take necessary precautions to prevent any potential disasters.

II. Literature review

Since the invention of smoke detectors, research and development in fire detection systems have improved significantly, giving way to increasingly sophisticated image- and video-based systems. Several image processing methods, including color-based methods, motion-based methods, pattern recognition, and deep learning, are used in fire detection, according to a thorough evaluation of the literature that has already been published. Although these techniques have proven effective in a variety of situations, accuracy, detection speed, and alerting should all be improved.

Several studies have suggested fire detection methods for video surveillance that make use of the OpenCV framework. For instance, Töreyin et al. (2006) suggested a system for flame detection in films that employs wavelet analysis and temporal color. A similar technique for detecting fires based on color, motion, and flicker was suggested by Celik (2010).

By improving the alert and notification process, our initiative seeks to solve shortcomings in earlier systems while building on these fundamentals. Our system not only detects potential fire dangers but also instantly notifies the user through email and an audible alarm. The system's dependability and response time are intended to be improved by the integration of computer vision and quick notification.

III. Proposed method

The many methods utilized to create an automatic fire detection and warning system are explained in this section. Python is the programming language used to create the project, and other modules are used to ensure its effective completion. OpenCV, NumPy, smtplib, playsound, threading, and Pygame are a few of these. We will go into great detail about the various functions and how they are implemented.

A. *Initialization*

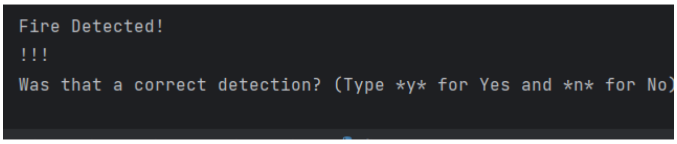
In order to track conditions like the status of the alarm, the status of the email, and the number of fires discovered, we declare a number of variables at the beginning of the code, including Alarm_Status, Email_Status, and Fire_Reported. Key parameter Sensitivity_Level is initially set to 15000 but is dynamically changed during configuration to improve detection precision.

B. *Alarm and Email Function*

'play_alarm_sound_function' and 'send_mail_function' are two key functions that are defined. When a fire is detected, the first function uses the Pygame module to play an alert sound. The second action notifies a specific recipient through email of a fire accident.

C. *Configuration and Sensitivity Adjustment*

The setup and calibration phase is started by calling the'start_configuration' function. This entails reading frames from two different video sources, scaling them, blurring one frame with Gaussian blur, and changing its color space from BGR to HSV. For the purpose of detecting fires, a color spectrum is established, and a mask is made and placed to the frame. When there are more red pixels than allowed, the system counts them and reports a fire. In order to improve system accuracy, sensitivity is automatically raised by 25% every 10 seconds if no fire is found.



```
Fire Detected!
!!!
Was that a correct detection? (Type *y* for Yes and *n* for No)
```

Fig. 1 If it is a false alarm, the senstivity is reduced.

D. *Fire Detection and Alert System*

Figures The fire detection system is effectively started after configuration when the'start_detector' function is called. The steps taken, which include frame reading, resizing, blurring, HSV conversion, and fire detection, are similar to those in the configuration stage. An alarm is activated and an email is sent if a fire is discovered.

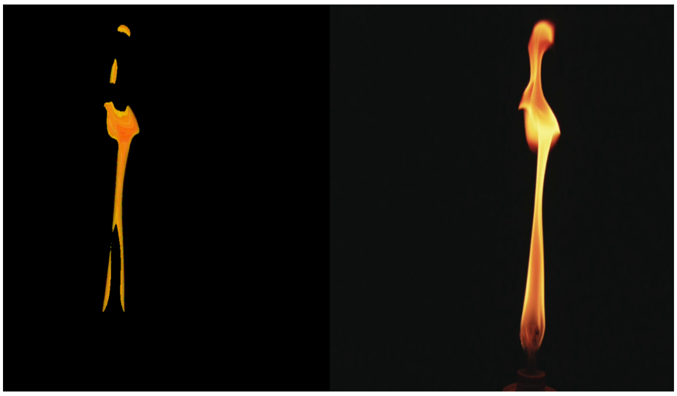
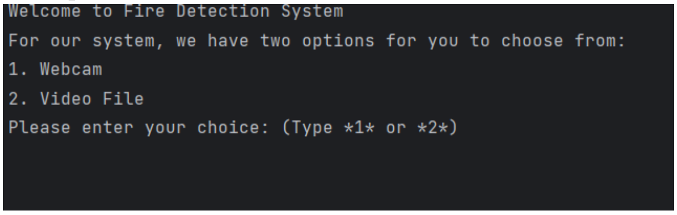


Fig. 2 original image on the right, masked image on the left

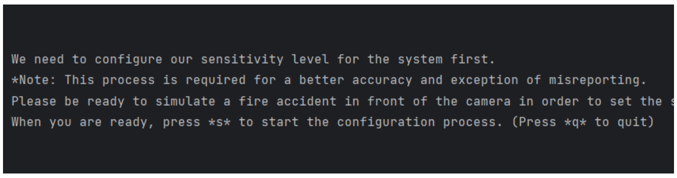
E. *User Interface and Input*

The system offers a straightforward user interface that asks users to select between video file input and webcam input. The user is prompted for the file path if the video file is chosen. In order to calibrate the system's sensitivity level, the user must first configure the system by simulating a fire. Real-time fire detection doesn't begin unless a configuration is accomplished.



```
Welcome to Fire Detection System
For our system, we have two options for you to choose from:
1. Webcam
2. Video File
Please enter your choice: (Type *1* or *2*)
```

Fig. 3 welcome the user



```
We need to configure our sensitivity level for the system first.
*Note: This process is required for a better accuracy and exception of misreporting.
Please be ready to simulate a fire accident in front of the camera in order to set the
When you are ready, press *s* to start the configuration process. (Press *q* to quit)
```

Fig. 4 information for the user

F. *System Exit and Cleanup*

Until the "q" key is pushed, the program continues to execute indefinitely. At that point, all OpenCV windows are closed and the video sources are released.

The core of the Automatic Fire Detection and Warning System is this suggested technique. This system is made to respond to possible fire dangers quickly and effectively, protecting the occupants and property and assuring safety. The system can do complex operations in real-time thanks to the usage of the Python language and associated libraries, making it a dependable and efficient response to fire emergencies. The modularity of the code also enables future updates and environment adaptability.

IV. Conclusions

Using the Python programming language, we created a "Automatic Fire Detection and Warning System Adaptable to Any System" for this project. The major goal of our research was to automatically detect fires using photos gathered from camera records or sporadic photographs, providing the user with quick alerts and email messages. We used a number of modules to implement the system, including cv2 (OpenCV), numpy, smtplib, playsound, and threading.

We concentrated on the following crucial elements during the development process:

1)Image Processing and Fire Detection: To process photos and identify flames, we made use of the robust capabilities of OpenCV and numpy. We developed a fire identification algorithm that locates areas with fire-like features by analyzing the image data. Accurate fire detection is made possible by the algorithm's ability to discern between typical scenarios and fire incidents.

2)Alarm and Notification: We implemented the "play_alarm_sound_function" and "send_mail_function" to swiftly notify users of fire events. Users are instantly informed even if they are not actively watching the system thanks to the 'play_alarm_sound_function', which activates an auditory alarm to draw rapid attention. The'send_mail_function' also notifies the user through email about the fire occurrence and includes extensive details in those emails.

3)System Optimization: We implemented optimization approaches to improve the effectiveness and performance of our system. For instance, in order to speed up processing while keeping a reasonable level of accuracy, we downsized the video frames. We achieved parallel processing by utilizing threading, enabling real-time fire detection and reducing delays.

We faced difficulties and complications throughout the project, including adjusting the fire detection algorithm, enhancing system performance, and managing probable false positive or false negative detections. We did, however, succeed in developing a trustworthy and efficient fire detection system through iterative development and rigorous testing.

Acknowledgment

We would like to thank our instructors and the entire department of computer engineering for their support and encouragement. They inspired us to strive for excellence via their commitment to creating a supportive learning environment.

We also want to thank our friends and classmates who helped us out and worked on the project with us throughout its many stages. Their comments and insights made a significant contribution to the project's growth and improved our comprehension of the topic.

.

References

[1] S. I. Thakur, P. Mishra, and S. Verma, "Real-time fire detection using computer vision and deep learning," in 2019 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS), 2019, pp. 1-5.

[2] S. A. Ghumare, M. R. Sable, and V. V. Patil, "Fire detection system using image processing techniques," in 2017 International Conference on Computing, Communication, Control and Automation (ICCUBEA), 2017, pp. 1-5.

[3] M. I. Hossain, M. G. Rasul, and I. Kabir, "Intelligent fire detection system using image processing," in 2018 International Conference on Networking, Systems and Security (NSysS), 2018, pp. 1-6.

[4] S. Singhal, R. Yadav, and A. Tripathi, "Fire detection using computer vision and Internet of Things," in 2018 International Conference on Computing, Power and Communication Technologies (GUCON), 2018, pp. 760-764..

[5] A. Jain, S. Bhandari, and S. Sengar, "Fire detection and alert system using image processing and machine learning," in 2020 International Conference on Communication and Signal Processing (ICCSP), 2020, pp. 0989-0993.