CASE STUDY On Fire Detection System using Computer Vision and Image Processing Technique

Technical Report · May 2019

DOI: 10.13140/RG.2.2.35663.74408

1 author:

Jaanvi Juneja
Symbiosis Institute of Computer Studies and Research

CASE STUDY
On
Fire Detection System using Computer Vision and Image Processing Technique

Submitted by

Jaanvi

17070122504

Under the Guidance of

Yogesh Kumar Dhawale
Programming Officer
SST, ONGC Ahmedabad

Department of Computer Science Engineering
SYMBIOSIS INSTITUTE OF TECHNOLOGY, PUNE

May 2019 – June 2019

ABSTRACT

Lately, fire outbreak is becoming an issue happening in ONGC oil well near Ahmedabad and the damage caused by these types of incidents is tremendous towards nature and human interest. After this incident, the need for an application for fire detection has increased. However, most of the available fire detection system uses temperature or smoke sensors which take time to respond. Moreover, these systems are costly and not effective if a fire is far away from the detectors. This led to thinking of alternatives such as computer vision, image processing techniques. One of the costs effective methods would be to use surveillance cameras to detect the fires in order to inform the relevant parties. The proposed research work suggests a method to use surveillance cameras in order to monitor occurrences of fire anywhere within the camera range. In this paper, a method has been used that finds the boundary of the moving region in the color segmented image and calculate the number of fire pixels in this area. Then a fire detection system is developed based on this method to detect fire efficiently to save life and property from fire hazard.

Contents

List of Figures

INTRODUCTION

Computer Vision based fire detection using image processing has the potential to be useful in conditions in which conventional methods cannot be adopted. The fire detection algorithm uses visual characteristics of fires like brightness, color, spectral texture, spectral flicker, and edge trembling to discriminate them from other visible stimuli. There are various fire detection techniques such as infrared sensor, a thermal detector, smoke detector, flame detector, and optical smoke detector. These methods are not always reliable as they do not always detect the fire itself but detect one or more phenomena resulting from fire, such as smoke, heat, infrared, ultraviolet light radiation or gas, which could be produced in other ways and hence, produces many false alarms. By the help of computer vision and image processing techniques, it is possible to get better results than conventional systems because images can provide more reliable information.

HAZARDS OF FIRE

"A fire broke out around 2030 hours on March 14 in an ONGC well at Nandej (Well #6) near Ahmedabad while workover (repair and maintenance) operations were being carried out,"

A late evening fire at a well of the ONGC Ahmedabad asset on Wednesday **claimed the lives of two contractual workers while four others sustained severe burn injuries**, official sources said on Thursday.

Read more at:
//economictimes.indiatimes.com/articleshow/68420144.cms?utm_source=contentofinterest&utm_medium=text&utm_campaign=cppst

Figure1 ONGC Fire

PROPOSED SYSTEM OVERVIEW

Computer vision-based detection system using color and motion properties of fire. Fire region detection approach tries to find the moving areas of the image. As in the real world, fire regions grows and spreads as the time passes. Finding the movement in the image reduces the candidate regions to detect fire color pixels. Contour detection is performed to find the moving regions of the image which is already segmented using color property. Then the boundary of the moving region is found and the amount of fire pixels in this area is calculated. Since fire region varies in a great extent, based on this approach can find whether there is really fire in the moving fire colored region or not.

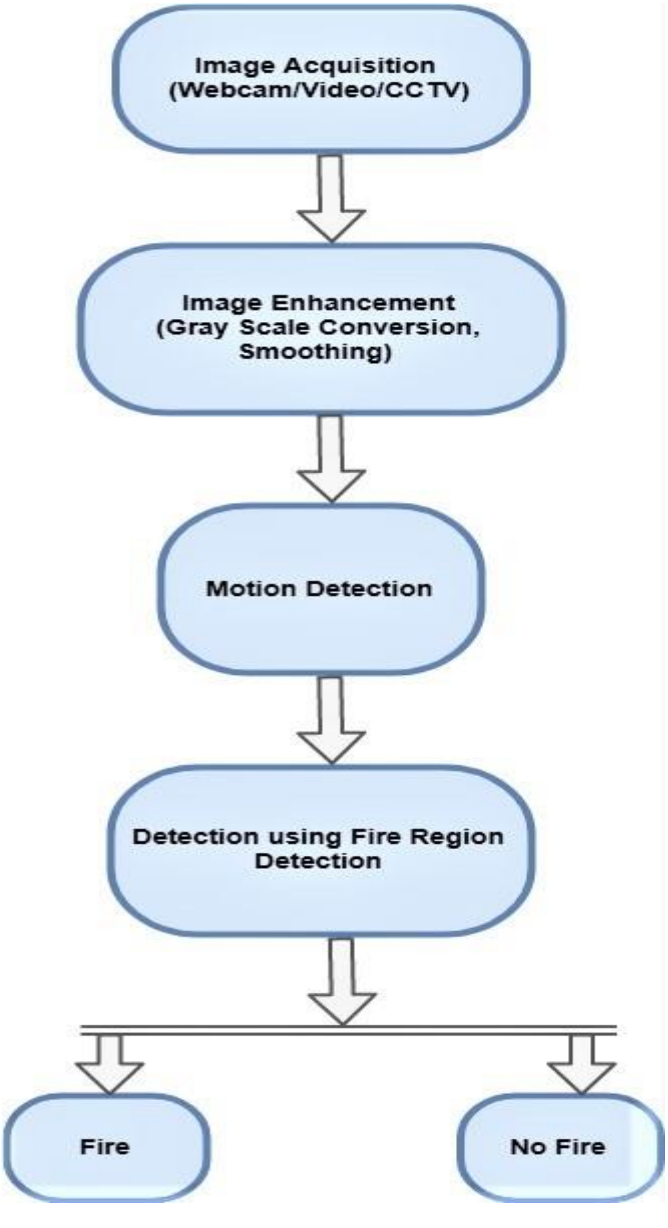

Figure2 Block Diagram

OBJECTIVES

The main objective of the project is to develop a fire detection technique in real time. The main points of the project objectives are given below:

- To detect fire while they are still small and have not grown too large.
- To develop a low cost fire detection system in real time.
- A system that detects a fire earlier than conventional fire detectors and more accurate.

SCOPE

The fire detection covers a very large field, and it would be impossible to cover all aspects in one project. The focus in this project is detection using a low-cost camera. This would mean that the program does not only work with expensive technology such as infrared cameras or other such cameras. The cameras that are required to at least work with this program are the CCTV cameras, such as those in shopping complexes or malls. One factor that needs to be taken into consideration is that, unlike other fire detectors, this system is not a point type detector. It should be able to detect fire in large open spaces, so that the whole scenario must be considered, and not just a single point on the image from the video feed. A system that can also be used in aggressive environments as well as in hazardous areas.

CHALLENGES

There have been so many challenges in this project. Working with different lighting condition was a challenge too. Another factor that created difficulty was when the color of the fire and some similar fire colored object overlapped because of light or for some other reasons. Detecting fire at initial stage in extreme sunny day is also a challenge. The biggest challenge was to complete the project and to increase its accuracy. This whole process is a very complex phenomenon and making a machine to interpret this as a human level has always been a challenge.

DESIGN AND IMPLEMENTATION

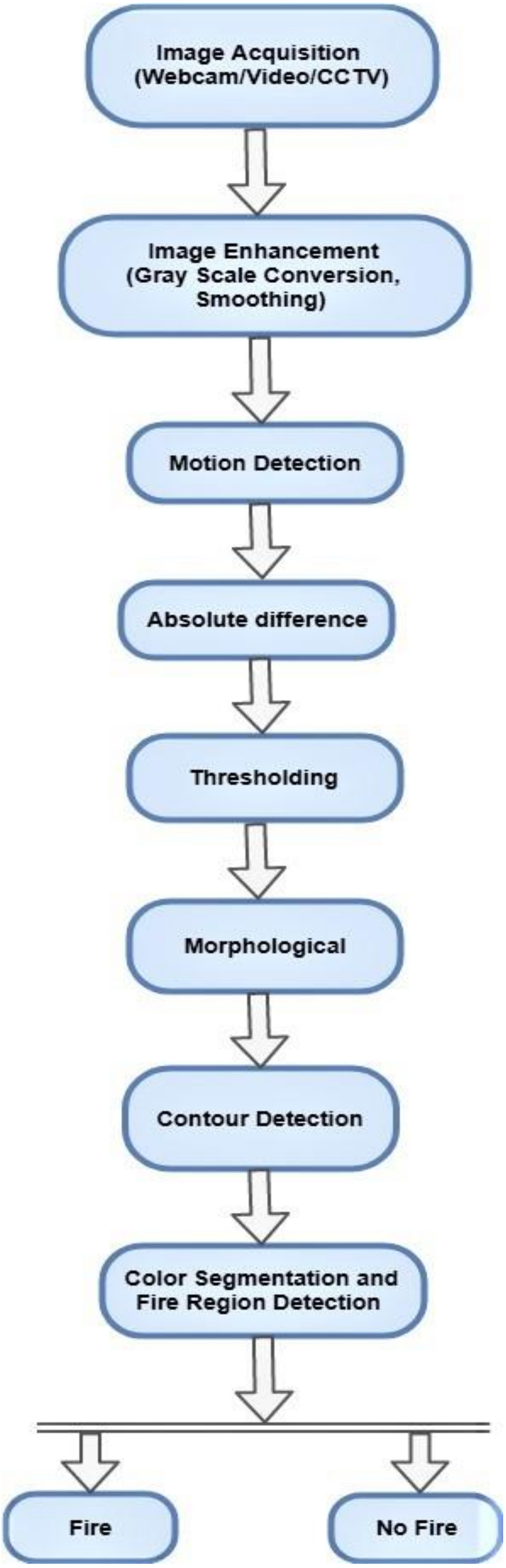

Figure 3 Block Diagram

Fire detection system using image processing

Step 1- Image Acquisition

At first the video camera is on and then the first frame captured is considered as initial background image. Then continuously frames are captured from the camera. The frame is taken in RGB color space.

The procedure is given below:

Import cv2

video = cv2.VideoCapture(0)

check, frame = video.read()

frame is a NumPy array, it represents the first image that video captures

check is a bool data type, returns true if python is able to read the videocapture object

Figure4 Input Image

Step 2- Gray Scale Conversion

All the images captured are converted into grayscale. Because in RGB format, the R, G, B values of each pixel differ from each other, but in gray level the R, G, B values are same. So, it becomes easier to compare two images in grayscale format rather than in RGB format.

The procedure is given below:

Convert RGB to grayscale.

grayImage = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

Step 3- Preprocessing- Preprocessed to enhance the image according to the specific task.

Preprocessing for-

- **Noise Reduction**
- **Brightness and contrast enhancement**

1) **Smoothing**

There are several situations, when image quality is reduced due to noise. So, before any further processing, in this step, noise cleaning of image is done. Smoothing is often used to reduce noise within an image or to produce a less pixelated image.

Convert the gray scale frame to **GaussianBlur.**

The procedure is given below:

Gray = cv2.GaussianBlur(gray,(21,21),0)

Figure5 GaussianBlur Image

Step 4- Motion Detection

System looks for if there is any motion in consequent frames. For this, background subtraction is used. Foreground objects result by calculating the difference between an image in the sequence and the background image (previously obtained). Then the movement of these foreground objects between successive frames are determined. By subtracting consequent frames, we find the area where motion occurred. The frames captured from camera is counted. First frame is considered as initial background frame. As time passes, background image is updated according to foreground image. There are several methods for separating foreground from the background. Since video is used, so some built-in function of OpenCV is used directly to achieve faster performance.

The procedure is given below:

Delta_frame = cv2.absdiff(first_frame,gray) #calculate the difference between the first frame and other frames

Figure6 Absolute difference between reference image and captured image

Thresh_delta = cv2.threshold(delta_frame, 30, 255, cv2.THRESH_BINARY)[1] #Black and white image

Thresh_delta=cv2.dilate(thresh_delta, None, iterations=0)

Figure7 Thresholding

Step 5- Morphological

Morphology is the process of object study and use to reduce noise. Sometimes, there can be local noisy high intensity changing with fire colors when there is no fire. To improve performance, erosion and dilation are applied. So, system can reduce noisy effects of mobile fire colored objects.

Dilation and Erosion

- Dilation increase white pixels in the image
- Erosion decrease black pixels in the image

The procedure is given below:

cvDilate(fire_image, fire_image, 0 ,1);

cvErode(fire_image, fire_image, 0, 1);

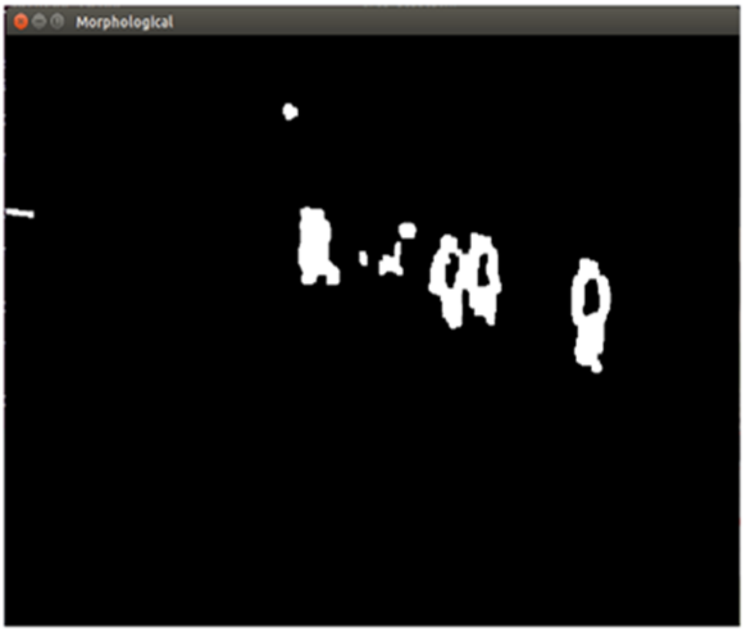

Figure8 Morphological

Step 6- Contour detection

Define the contour area. Basically, add the borders.

The procedure is given below:

```
(_,cnts,_)=cv2.findContours(thresh_delta.copy()
,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)
for contour in cnts:
    if cv2.contourArea(contour)<1000:
```

```
        continue
    print('1')
    (x, y, w, h)= cv2.boundingRect(contour)
    cv2.rectangle(frame, (x,y), (x+w,y+h),(0,255,0),3)
```

Figure9 Contour Detection

Source Code of motion detection

```
import cv2, time

first_frame = None

video = cv2.VideoCapture(0)

while True:
    check, frame = video.read()
    gray = cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)
    gray = cv2.GaussianBlur(gray,(21,21),0)

    if first_frame is None:
        first_frame = gray
        continue

    delta_frame = cv2.absdiff(first_frame,gray)
    thresh_delta = cv2.threshold(delta_frame, 30, 255, cv2.THRESH_BINARY)[1]
    thresh_delta = cv2.dilate(thresh_delta, None, iterations=0)
    (_,cnts,_) =
cv2.findContours(thresh_delta.copy(),cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_
SIMPLE)
    for contour in cnts:
```

```
        if cv2.contourArea(contour)<1000:
            continue
        print('1')
        (x, y, w, h)= cv2.boundingRect(contour)
        cv2.rectangle(frame, (x,y), (x+w,y+h),(0,255,0),3)

    cv2.imshow('frame',frame)
    cv2.imshow('capturing',gray)
    cv2.imshow('delta',delta_frame)
    cv2.imshow('thresh',thresh_delta)
    key = cv2.waitKey(0)
    if key== ord('q'):
        break

video.release()
cv2.destroyAllWindows()
```

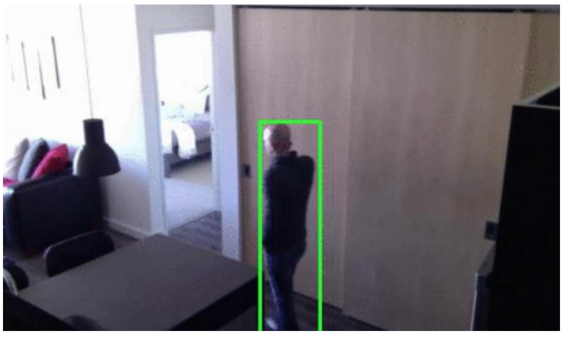

Figure10 OUTPUT

Step 7- Color Segmentation and Fire Detection

Finding Fire Pixels

Converting grayscale image into RGB image.

```
Backtorgb=cv2.cvtColor(gray,cv2.COLOR_GRAY2RGB)
```

To find possible fire pixels, firstly we find fire colored pixels according to RGB values of video frames. We used following RGB values to detect fire:

R>220 G>200 125<B<185

R>220 125<G<150 75<B<100

R>220 175<G<225 75<B<125

Most of the fire colored pixels is in these three ranges. So, if one pixels RGB values are in these ranges, it is fire colored pixel. Another property to look is fire pixels have a relation among its color channels that is : R > G > B.

Now it is time to check consecutive frames for fire regions. If it is found that there are enough candidates in consequent frames, then it can be said that fire is detected.

STRENGTHS AND WEAKNESS OF THE SYSTEM

Strengths

Significantly has the quicker response time.

usage in aggressive environments (chemically environments) as well as in hazardous areas.

No addition hardware components required.

The system uses a low-cost camera to capture images in order to determine whether fire exists in the video caught by the camera. This method therefore has all the weaknesses and strength associated with a machine vision system.

Weakness

• One of the weaknesses in this system is that it cannot detect what it cannot see. If the fire exists in the range of the video which is being blocked by an object, the system will not be able to detect it.

• The method used also has its weaknesses, as can be seen from the results. It gives false alarm in some cases when there is no fire but an extreme fire colored object exists.

FUTURE WORK

Future works to overcome the limitations as stated before are described below:

• To develop a robust fire detection system which gives less false alarm in case of extreme fire colored object.

• To develop a combined fire and smoke detection

References

OpenCV Python Tutorials

https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_tutorials.html

Digital Image Processing Tutorials

https://www.tutorialspoint.com/dip/