International Journal of
Human Computations & Intelligence
Vol. 03, Issue. 03, May/June 2024

RESEARCH ARTICLE OPEN ACCESS

# Smart Surveillance system using Gaussian Mixture Model

**Nihaal Abdul Baseer . Mohammed Samiuddin . MD Imran . Ankur Deb . Naveen Chandra Gowda**

School of Computer Science and Engineering, REVA University, Bengaluru, India.

DOI: **10.5281/zenodo.13933443**
Received: 20 February 2024 / Revised: 11 March 2024 / Accepted: 23 May 2024


**Abstract –** Surveillance systems are in huge demand and at best, a necessity in developed and developing communities. But with the rise in demand, a basic integrated system costs a significant amount. This is my attempt at creating an intruder detection system with the tools that are easily accessible and understanding and choosing between the different approaches to solving this problem. We also look at different ways you can better this to your personal preferences. As is, this is a perfect implementation for a general plug-and-play use case.

**Index Terms –** Surveillance systems, Gaussian Mixture Model

## I. INTRODUCTION

The need for an intruder detection/surveillance system has been amplified in the recent years, with many belongings and places of value being unattended. Any basic system has a complex way of setting up and integrating into a household and many systems aren't compatible with the existing smart system of a household despite costing a lot. Not to mention most surveillance systems aren't intruder detection systems to begin with. A premier intruder detection system costs fortune having the need for various sensors and costly cameras [1]. Taking this all into consideration, there are three main aspects that need to be addressed to make this system viable and convenient for usage: Cost of implementation, Accessibility and Portability.

## II. LITERATURE SURVEY

In reference to the previous research that has been done on this topic, there were 3 main components that were integral for the successful working of the system [2][3]. Establishing a connection to a live or pre-recorded video feed through a camera. Segmentation of the background with the foreground. Choosing the right algorithm to perform the separation. Before this approach of background separation, the recurring problem was of the unwanted elements in the image and unnecessary disturbances. Precise

MILESTONE RESEARCH.IN OPEN ACCESS

ISSN (Online): 2583-5696
Int. Jr. of Hum Comp. & Int.

object segmentation is essential for surveillance video data, both indoors and outdoors. However, a number of issues, such as shifting lighting, noise from the camera, reflections, and dynamic motion patterns, make this task more difficult [4][5]. It is possible for static and dynamic objects to coexist in the video, each posing different segmentation challenges. For surveillance systems to be effective, they must be able to overcome these obstacles so that they can consistently identify and track objects of interest in spite of changes in the surrounding area and possible disruptions in the video feed [6][7].

But a deeper look at the problem yields the solution itself. Instead of treating the background as hostile, treat it as a part of the base foundation of reference. From there we observe the differences in the frame, through a stated period of time [8][9]. And classify the range of different changes that are caused in the frame from different objects and classify them into unnecessary changes (ex. Branches shaking, brightness changes due to lighting, etc.) and observable and notable changes (ex. A person entering the frame, a car moving along the road, etc.). Keeping this in mind my model works based on recognizing a base frame and referencing it with the current frame [10].

## III. PROPOSED SOLUTIONS

A technique based on the Gaussian Mixture Model is used to overcome problems encountered by other methods in the segmentation of videos. It is assumed that pictures of the scene devoid of obtruding objects show a predictable behavior that a statistical model can adequately capture. Once a statistical model of the scene has been created, the parts of the image that deviate from the model can be used to detect the presence of intruding objects. This method is often called "background subtraction." This technique uses the Gaussian Mixture Model to separate any anomalies introduced by moving or dynamic objects from the expected scene background, with the goal of improving the accuracy of object segmentation in surveillance videos.

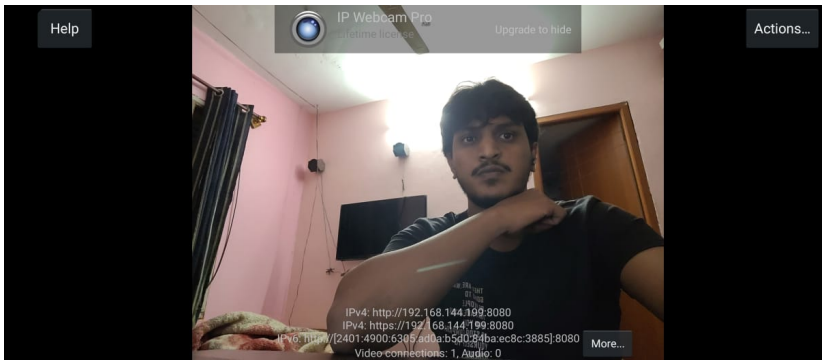
So by segmenting the existing frame and analysing the patterns of the existing frame, we can detect any changes and classify them as an intrusion and to add to the adaptability of the program, through the means of the Gaussian Mixture Model, we will be able to analyse the scene and depending on the unnecessary noise, change the threshold values and sensitivity to changes to be a more effective and adaptive way to solve this problem. *With its ability to automatically calculate the number of components needed for each pixel, the new algorithm can seamlessly adjust to the scene being observed. This results in a marginal improvement in segmentation accuracy in addition to a reduction in processing time. The algorithm improves efficiency and adaptability by dynamically varying the number of components per pixel according to the scene's features, which leads to an overall improvement in segmentation performance.*

Now, to address the 3 aspects that make this system viable, we need a cost effective implementation. For that the major issue that needs addressing is a camera to record feed. For that purpose we shall use a remote webcam to set up as a link to our server. Secondly, we need a way to address a way to import the live webcam feed to our program to assess without having a very resource draining application. Lastly, we need a way to access our feed from anywhere and alert the owner of any intrusion detected without the need of a specific app, as it is one more additional step than using a pre-existing messaging service.

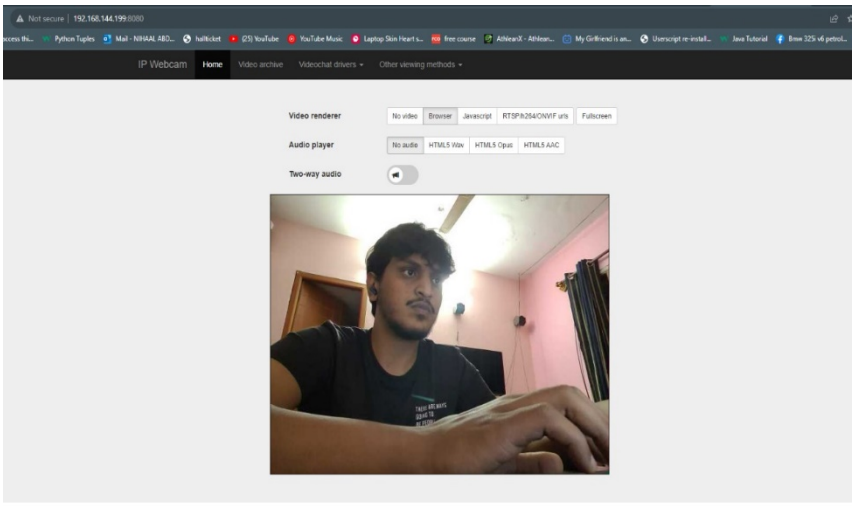
## IV. IMPLEMENTATION AND RESULTS

- First step is to establish an input of a live camera feed. For this I use a basic webcam service that turns any mobile device or laptop with a camera into a remote live recording device.

- After the installation of the app: IP Webcam, you can start a feed and observe the interface to be something similar to this



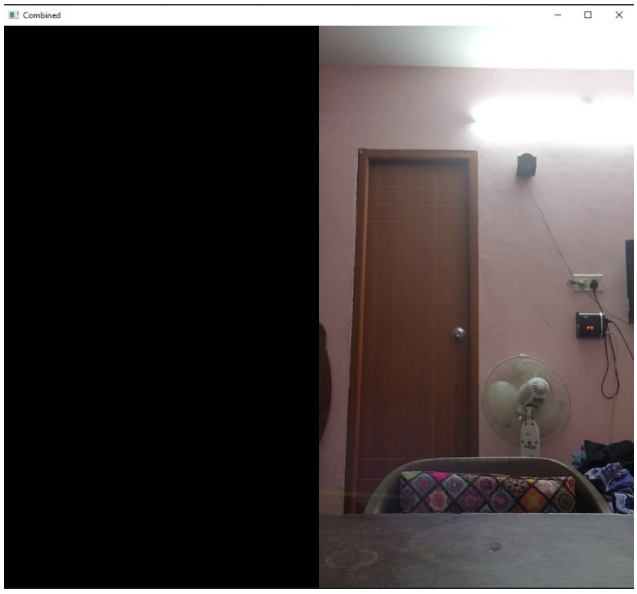

- You need to copy the ipv4 address shown at the bottom of the video feed to be able to analyse it in the program. Which in this case is : 192.168.144.199:8080
- Now for the purpose of basic surveillance this feed can be accessed through any device by typing in the ip address in any browser.



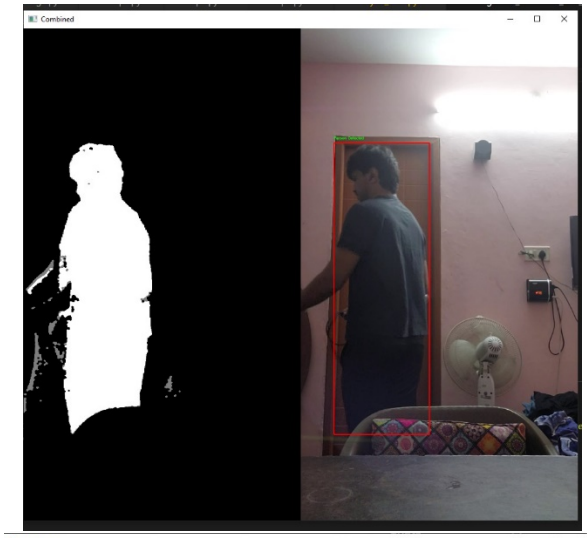

- But we need to analyse it for any intrusion detection even when it isn't being physically monitored through web browser by a person. It needs to be a constantly running service.
- For that purpose we take the video feed and paste it in our code, like demonstrated:
- "`cv2.VideoCapture(protocol://username:password@host:port/video)`"
- For example:

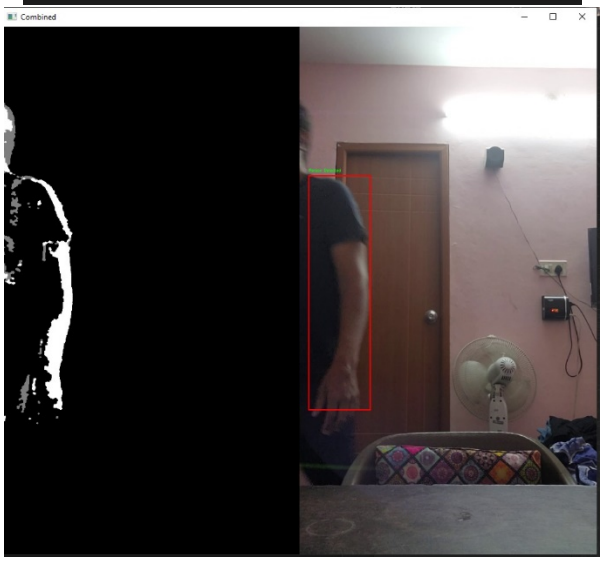
```
import numpy as np
import cv2
import time
import datetime
from collections import deque

cap = cv2.VideoCapture('http://192.168.144.199:8080/video')

# you can set custom kernel size if you want
kernel = cv2.getStructuringElement(cv2.MORPH_ELLIPSE, (5,5))

# initilize background subtractor object
bgsub = cv2.createBackgroundSubtractorMOG2(detectShadows=True)

while(1):
    ret, frame = cap.read()
    if not ret:
        break

    # Apply background subtraction
    fgmask = bgsub.apply(frame)

    # Apply some morphological operations to make sure you have a good mask
    fgmask = cv2.morphologyEx(fgmask, cv2.MORPH_OPEN, kernel)
    fgmask = cv2.morphologyEx(fgmask, cv2.MORPH_CLOSE, kernel)

    # Detect contours in the frame
    contours, hierarchy = cv2.findContours(fgmask,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)

    if contours:

        # Get the maximum contour
        cnt = max(contours, key = cv2.contourArea)


        # make sure the contour area is somewhat higher than some threshold to make sure its a person and not some noise.
        if cv2.contourArea(cnt) > 1000:

            # Draw a boundjng box around the person and label it as person detected
```

- To demonstrate the working of the background subtraction , I shall use a sample recording from the video captured
- Following is a demonstration of how a static scene is recognised without any intrusion:




- Now when any change is detected in this frame it is recognised like so:

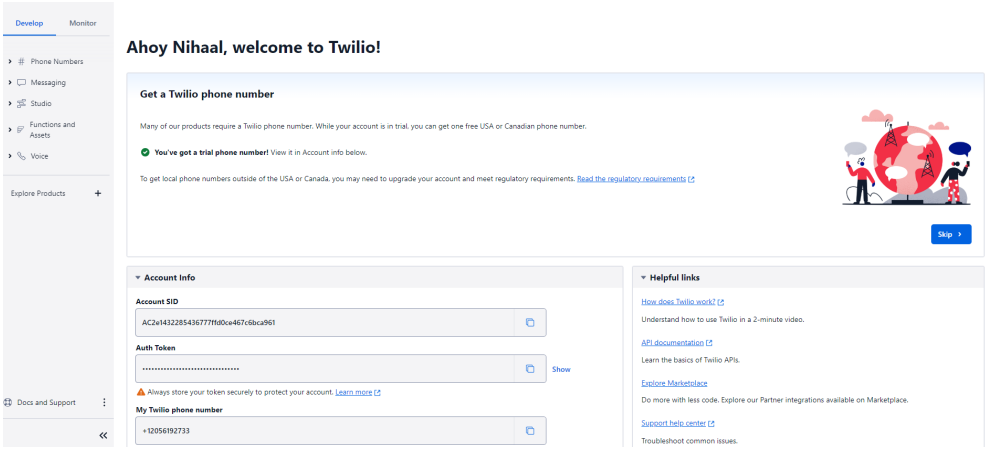


- The demonstration for this is done using this code snippet:

```python
import numpy as np
import cv2
import time
import datetime
from collections import deque

cap = cv2.VideoCapture('http://192.168.144.199:8080/video')

# you can set custom kernel size if you want
kernel = cv2.getStructuringElement(cv2.MORPH_ELLIPSE, (5,5))

# initilize background subtractor object
bgsub = cv2.createBackgroundSubtractorMOG2(detectShadows=True)

while(1):
    ret, frame = cap.read()
    if not ret:
        break

    # Apply background subtraction
    fgmask = bgsub.apply(frame)

    # Apply some morphological operations to make sure you have a good mask
    fgmask = cv2.morphologyEx(fgmask, cv2.MORPH_OPEN, kernel)
    fgmask = cv2.morphologyEx(fgmask, cv2.MORPH_CLOSE, kernel)

    # Detect contours in the frame
    contours, hierarchy = cv2.findContours(fgmask,cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_SIMPLE)

    if contours:

        # Get the maximum contour
        cnt = max(contours, key = cv2.contourArea)


        # make sure the contour area is somewhat higher than some threshold to make sure its a person and not some noise.
        if cv2.contourArea(cnt) > 1000:

            # Draw a bounding box around the person and label it as person detected
            x,y,w,h = cv2.boundingRect(cnt)
            cv2.rectangle(frame,(x ,y),(x+w,y+h),(0,0,255),2)
            cv2.putText(frame,'Person Detected',(x,y-10), cv2.FONT_HERSHEY_SIMPLEX, 0.3, (0,255,0), 1, cv2.LINE_AA)


    # Stack both frames and show the image
    fgmask_3 = cv2.cvtColor(fgmask, cv2.COLOR_GRAY2BGR)
    stacked = np.hstack((fgmask_3,frame))
    cv2.imshow('Combined',cv2.resize(stacked,None,fx=0.65,fy=0.65))

    k = cv2.waitKey(40) & 0xff
    if k == ord('q'):
        break

cap.release()
cv2.destroyAllWindows()
```

- Now for the part where we alert the user of the intrusion, we use an online sid generator to send an SMS to a registered phone number.
- The service that we'll be using is Twilio for which python has a supported library.
- After the generation of the account in twilio the dashboard should look like this:



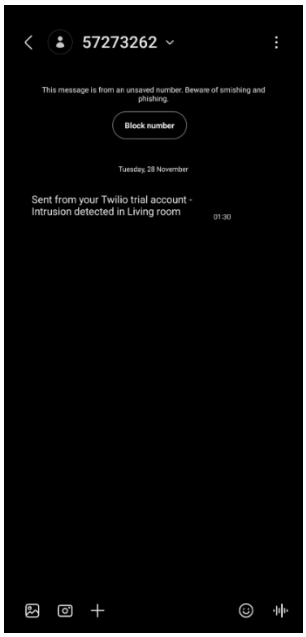

- Now select the account sid and authorisation token from the website and paste in the code like so:

```python
import cv2
import time
import os
import shutil
import json
import argparse
import datetime
import twilio.rest

def read_credentials(file_path):
    with open('credentials.txt', 'r') as file:
        credentials = json.load(file)
    return credentials

def send_sms_alert(credentials):
    client = twilio.rest.Client(credentials['AC2e1432285436777ffd0ce467c6bca961'], credentials['c2d7812a1d3d5d93b332ae32b59aa169'])

    message = client.messages.create(
        body="Unauthorized Person detected in your room",
        from_=credentials['12056192733'],
        to=credentials['918123841999']
    )

def detect_person(frame):
    # load the pre-trained Haar cascade for person detection
    person_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

    # convert the frame to grayscale
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    # perform person detection
    people = person_cascade.detectMultiScale(gray, scaleFactor=1.1, minNeighbors=5, minSize=(30, 30))

    # If person detected
    if len(people) > 0:
        return True
    else:
        return False

# Define the path to save images
current_path = os.getcwd()
capture_path = os.path.join(current_path, "capture")

if not os.path.exists(capture_path):
    os.makedirs(capture_path)

# Load the credentials from the text file
credentials = read_credentials('credentials.txt')

# Start the video capture
cap = cv2.VideoCapture(0)

while True:
    # Capture frame-by-frame
    ret, frame = cap.read()

    # Detect if a person is present in the frame
    if detect_person(frame):
        send_sms_alert(credentials)

        # Capture images every 3 seconds until no person is detected
        while detect_person(frame):
            cv2.imwrite(os.path.join(capture_path, f"{datetime.datetime.now()}.jpg"), frame)
            time.sleep(3)
            ret, frame = cap.read()

    # Display the resulting frame
    cv2.imshow('Intrusion Detection', frame)

    # Exit the program if the user presses 'q'
    if cv2.waitKey(1) & 0xFF == ord('q'):
```

- From here you can configure the message that is being sent to the phone on detection of any intrusion.
- Furthermore, to demonstrate the credibility of Twilio you can use the following code snippet to verify of its working:

```python
from twilio.rest import Client

# Read text from the credentials file and store in data variable
with open('credentials.txt', 'r') as myfile:
  data = myfile.read()

# Convert data variable into dictionary
info_dict = eval(data)

# Your Account SID from twilio.com/console
account_sid = info_dict['account_sid']

# Your Auth Token from twilio.com/console
auth_token  = info_dict['auth_token']

# Set client and send the message
client = Client(account_sid, auth_token)
message = client.messages.create( to =info_dict['your_num'], from_ = info_dict['trial_num'], body= "Intrusion detected in Living room")
```

- Results in the following output in the receivers device:




## V. CONCLUSION

To summarise from the findings of this project, I can conclude that I have yielded a satisfactory result in attaining a sustainable yet robust intrude detection/surveillance system. The use of the GMM model seems to be an appropriate fit as it provides for easy altering of the threshold values and adjust according to the surroundings. The changes that can be made to improvise on this are completely based on personal use cases, where in if you are using a single device for a location, you can assign a static ip so as to not need to alter the ip in the program every time you start a new server to send the live video feed. You can also use twilio to further your use case and send messages directly through whatsapp and also the recorded video of the intrusion using the VideoWriter() in pythons OpenCV. Overall it is a robust plug-and-play system for an intruder detection system.

## REFERENCES

[1] Zivkovic, Z., & Van Der Heijden, F. (2006). Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern recognition letters*, *27*(7), 773-780.

[2] Srinivasan, K., Porkumaran, K., & Sainarayanan, G. (2009, August). Improved background subtraction techniques for security in video applications. In *2009 3rd International Conference on Anti-counterfeiting, Security, and Identification in Communication* (pp. 114-117). IEEE.

[3] Samuel D Jonathan, Jonathan S Paul, Naveen Chandra Gowda, Ambika B J, & Kiran Kumar P N. (2023). Comparative Analysis of Cryptographic Algorithms. *International Journal of Human Computations & Intelligence*, *2*(5), 212–219. https://doi.org/10.5281/zenodo.8068102

[4] Garcia-Garcia, B., Bouwmans, T., & Silva, A. J. R. (2020). Background subtraction in real applications: Challenges, current models and future directions. *Computer Science Review*, *35*, 100204.

[5] Sudhanva Manjunath, Athreya Abhay Pratap Singh, Naveen Chandra Gowda, Yerriswamy T, & Veena H N. (2023). Machine Learning Techniques to Detect DDoS Attacks in IoT's, SDN's: A Comprehensive Overview. *International Journal of Human Computations & Intelligence*, *2*(4), 203–211. https://doi.org/10.5281/zenodo.8027034

[6] Zivkovic, Z. (2004, August). Improved adaptive Gaussian mixture model for background subtraction. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.* (Vol. 2, pp. 28-31). IEEE.

[7] Braham, M., Piérard, S., & Van Droogenbroeck, M. (2017, September). Semantic background subtraction. In *2017 IEEE International Conference on Image Processing (ICIP)* (pp. 4552-4556). Ieee.

[8] Xia, Y., Ning, S., & Shen, H. (2010, May). Moving targets detection algorithm based on background subtraction and frames subtraction. In *2010 The 2nd International Conference on Industrial Mechatronics and Automation* (Vol. 1, pp. 122-125). IEEE.

[9] Ananya B L, Nikhitha V, S Arjun, & Naveen Chandra Gowda. (2023). Survey of applications, advantages, and comparisons of AES encryption algorithm with other standards. *International Journal of Computational Learning & Intelligence*, *2*(2), 87–98. https://doi.org/10.5281/zenodo.7921019

[10] Ahmed, S. T., & Basha, S. M. (2022). *Information and communication theory-source coding techniques-part II*. MileStone Research Publications.

[11] Fathima, A. S., Basha, S. M., Ahmed, S. T., Mathivanan, S. K., Rajendran, S., Mallik, S., & Zhao, Z. (2023). Federated learning based futuristic biomedical big-data analysis and standardization. *Plos one*, *18*(10), e0291631.

[12] Basha, S. M., Ahmed, S. T., Iyengar, N. C. S. N., & Caytiles, R. D. (2021, December). Inter-locking dependency evaluation schema based on block-chain enabled federated transfer learning for autonomous vehicular systems. In *2021 Second International Conference on Innovative Technology Convergence (CITC)* (pp. 46-51). IEEE.

[13] Manzanera, A., & Richefeu, J. C. (2007). A new motion detection algorithm based on Σ–Δ background estimation. *Pattern Recognition Letters*, *28*(3), 320-328.