iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 10 **Issue:** VII **Month of publication:** July 2022

DOI: https://doi.org/10.22214/ijraset.2022.45854

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 10 Issue VII July 2022- Available at www.ijraset.com

Design and Development of Optimized Video Stream processing for Surveillance

Amarnath Swami[1], Rashmi Patil[2], Indira Umarji[3], Dr Vidyagowri Hemadri[4], Dr U P Kulkarni[5]
[1]*Student,* [2, 3, 4]*Assit Professor,* [5]*Professor, Dept of Computer Science and Engineering, SDM College of Engineering and Technology, Dharwad, Karnataka, India*

Abstract: ***The majority of current surveillance systems require human participation, with security personnel watching the monitor system & constantly spotting security breaches brought on by operator distraction or the potential for missing persons being discovered. Due to these difficulties, automated video surveillance is required in order to track, detect, and record security vulnerabilities in a monitored environment. We have reviewed the automation of monitoring using IP cameras in this paper. There are many uses for video surveillance. With the usage of Kafka, we have concentrated primarily on live streaming of video surveillance, face detection, and motion/object identification.***
Keywords: Track, security vulnerabilities, detect, face detection

I. INTRODUCTION

Continuous streaming of video streams from a host to a clients is known as video streaming. Users can watch videos online without downloading and installing them due to video streaming. Videos, TV shows, Videos online, and live streamed content are all examples of streaming video content. Video streaming to members has been a huge success for services like Netflix any communication buffer that publishes messages independent of their consumption serves as a repository for in communications. The storage system Kafka is excellent. For failure tolerance, information transferred to Kafka is copied to memory. Producers are permitted to wait for acknowledgement in Kafka. A write isn't finished until it's fully duplicated and assured to last if the server it was written to fails. Kafka can scale effectively because of the disc structures it employs. Even if you have 50 KB or 50 TB of data storage on the server, Kafka functions the same.

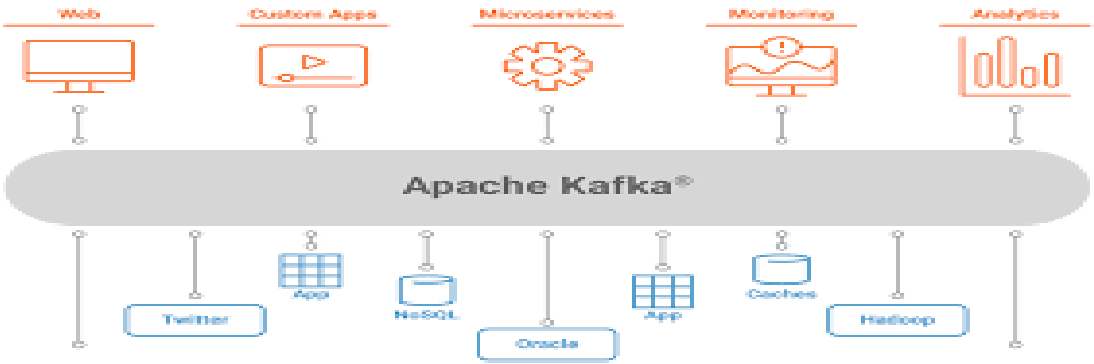
As a distributed file system with a specific focus on high-performance, low-latency commit data storage, replication, and propagation, Kafka enables the client to control their read position.

1) Publishes and receives subscriptions to the records streams, much as an enterprise the broadcast message or message queue..
2) Stores records in streams in a reliable, fault-tolerant manner.
3) Streams of records are processed as they stream out.

On one or even more hosts, Kafka will run as a cluster that can handle numerous data entry points. In groups called topics, the Kafka cluster maintains streams of records. Each record has a timestamp, a unique id and a key. Kafka has these fundamental APIs.

Kafka Producer API: Applications can deliver data streams to the Kafka cluster using the Kafka Producer API. Applications can read data streams from the cluster using the Kafka Consumer API.

II. MODELING AND ANALYSIS



A. Kafka Consumer API

Applications have the ability to connect to topics and analyse the records delivered to them in a stream.

B. Kafka Connector API

Applications are capable of acting as stream processors, converting input streams from one maybe more topics into output streams by consuming input streams from those topics and producing output streams for one or many output topics.

Create and operate scalable producer or consumers that link Kafka topics to already-running programmes or information systems. A link to a relational database, for instance, may record each modification made to a table. In Kafka, clients and servers communicate with one other via the straightforward, high-performance, and language-neutral TCP protocol. This protocol has a versioning system and keeps earlier versions compatible. Kafka comes with a Java client, but there are also clients for many other languages.

Topics and Logs:The topic is the fundamental abstraction that Kafka offers for just a stream of records.A topic is a name given to a category or stream where records are published. Kafka's topics always have multiple subscribers. It follows that a topic may have none, one, or many subscribers to the information written to it.

C. IP Camera

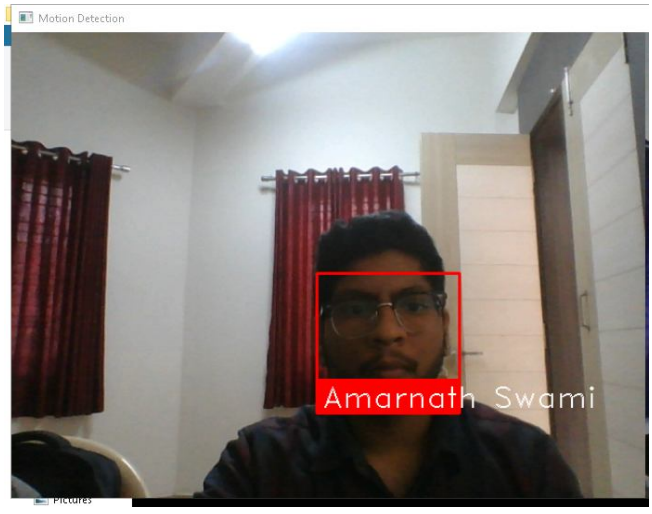
Digital video cameras that accept control data and transmit image information via an IP network are known as IP cameras. They are frequently employed for surveillance but, unlike analogue shuttered television (CCTV) cameras, only a local area network is needed. Although the majority of IP cameras are webcams, the phrase "IP camera" or "netcam" often only refers to those that can be accessed directly over a network connection. Internet Expert Network cameras are created for both commercial and residential applications. Home security consumer IP cameras often stream live footage to a chat feature on the user's device. They often use Ethernet cables or Wi-Fi to connect to the internet. Enterprise IP Cameras, in contrast to consumer IP Cameras, frequently provide higher picture quality, video analytics, and are accessed primarily through HTTP and real-time streaming protocol (RTSP).

D. Facial Recognition System

An illustration of biometrics is facial recognition, which recognises a human face by examining facial features in a photograph or video. If a camera enables users to create a database of family members and friends, the system should be able to determine whether a particular person is present in the database. If the camera has an accurate facial recognition feature, it can determine whether the subject detects is authorised (in the database). The owner might contact law enforcement if they find uninvited people. It is possible to identify and catch perpetrators using the video.

E. Algorithm To Install Facial-Recognition

1) *Step1:* Import and install the face-recognition packages pip install face-recognition
2) *Step2:* Load the images to the face-recognition package
3) *Step3:*Give the path locations for the face recognition package
4) *Step4:* Known and unknown images of the person will be displayed on the monitor as result



F. How To Stream Your IP Camera To Your Server From Anywhere In The World

Imagine that your camera can quickly and easily send what it recognizes to your server in real time. Certainly, this will simplify things for us to view the events captured on your CCTV, even if it has been destroyed or stolen. Of course, this technique is only applicable to IP cameras, which don't require an NVR or DVR to save CCTV footage. Nowadays, most cctv IP cameras come with a microSD memory card for video storage. You will, however, lose important proof if someone with malicious motives breaks or removes your microSD camera. Unfortunately, not everyone considers using the cloud to store video backups. therefore, a quick technique from me that could be created with the use of Python and OpenCV.

In this project, we'll use the decentralized platform for data streaming known as Kafka Message Brokers to communicate image data from an IP camera. In essence, A publish/subscribe communication system is Apache Kafka, where one or more systems provide data for a specific topic in Apache Kafka in real time (referred to as Producers). The topic can then be retrieved between one or more systems that require real-time data from the topic (referred to as Consumers)

Step1: Import and install Install libraries

OpenCV is a Python fully accessible toolkit used for face recognition, machine learning, and other computer vision applications. etc

Step2: Then, we may set up Apache Kafka.

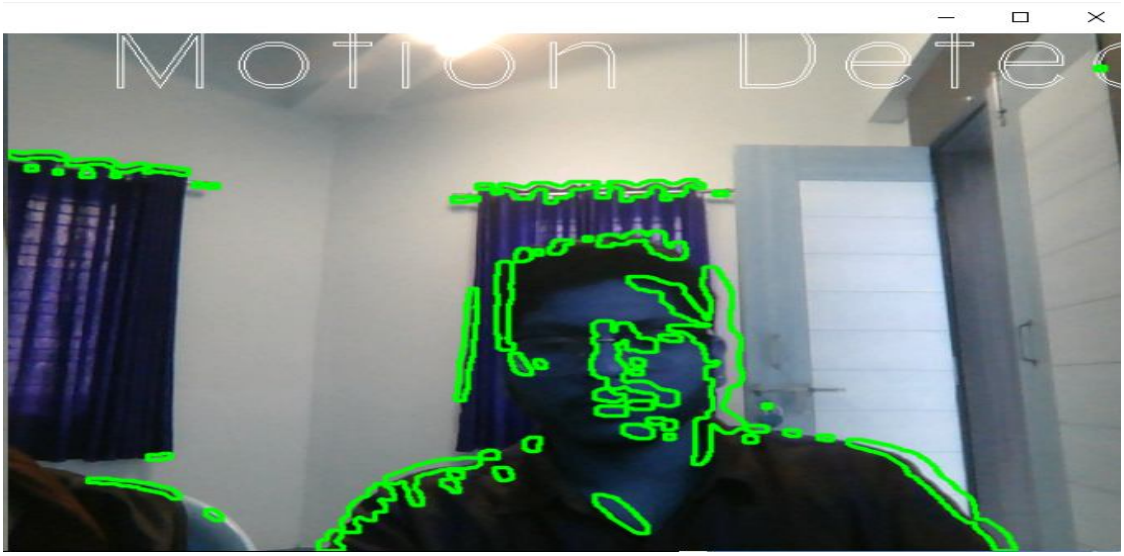
obtain the rtsp link and set up the camera: **we use onvif** Providing and advancing standardized interfaces for efficient compatibility of Internet protocol based physical security mechanisms is the mission of the open industry forum known as ONVIF.

In networks for entertainment and communication, streaming media servers are managed by the Real Time Streaming Protocol (RTSP). Media streams between end points are established and managed using the protocol. In order for the process to be controlled in real-time and occurs from the host to an user or by a user to server, clients of media servers send VHS-style instructions, include pause, capture, and play (Voice Recording).

G. Detecting Motion With Opencv — Image Analysis

Motion detection serves a variety of needs. When you notice activity on a security camera or a wildlife camera, for example, you can use this to begin recording. Performance optimization is another use. We simply need to examine a few moving, small portions of an image rather than the entire thing. similar to merely noticing the colour of the moving object.

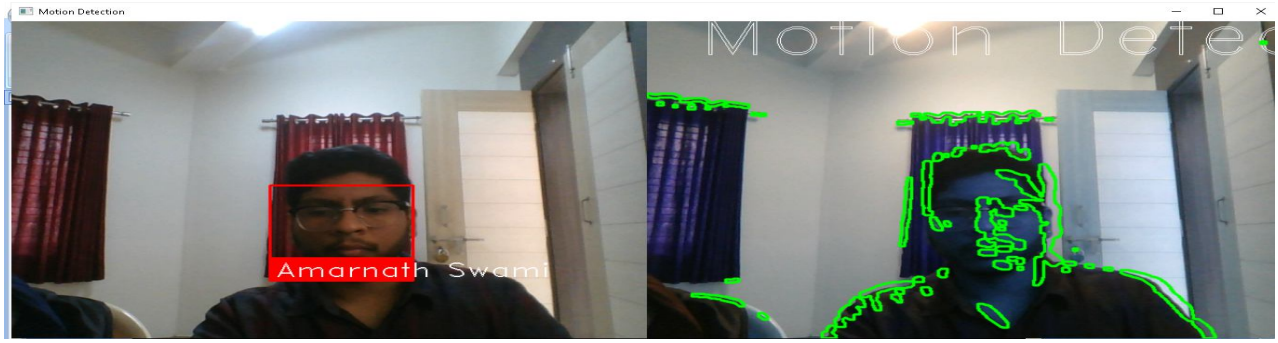
The analyst has some pretty sophisticated models for identifying whether an image is of a person, a bike, a bus, or a car, but they are unable to apply these models to the complete image. It is our responsibility to prepare the image. We must collect brief samples on which to test the identification model. How do we start recognizing motion? In order to identify any areas that have changed, we will examine each frames of a livestream through my camera to the previous one. The final outcome will resemble this:



H. Creating The Motion Detector

Reading up and getting our frame ready, We'll try reading the image first, then convert it from BGR to RGB using OpenCV's default RGB colors.

The next step is more exciting, though; we make the image grayscale and blur it to make it a little smoother. All RGB pixels are converted to a range between 0 and 255, whereby 0 is black & 255 is white, when they are converted to grey. Compared to handling three values, this is substantially faster (R, G and B). This is what happened:



Determined action (change compared to the previous frame)We'll perform the motion detection in this section. By comparing the pixel values, we may compare the previous and current frames. Keep in mind that every pixel is expressed by a single number from 0 and 255 as the image has been transformed to grey.

Not each pixel, but the zone that has altered since the previous frame is what we're looking for. Finding a location is the first step in accomplishing this. The function cv.findContours retrieves the contours or outside bounds from each white dot in the portion above. All contours are located and drawn in the algorithm below

1) *Step1:* import all the previous frames using loop and continue
2) *Step2:* calculate difference and update previous frame
   diff_frame = cv2.absdiff(src1=previous_frame, src2=prepared_frame)
   previous_frame = prepared_frame
3) *Step3:* Dilute the image a bit to make differences more seeable; more suitable for contour detection
4) *Step4:* Only take different areas that are different enough (>20 / 255)
   thresh_frame = cv2.threshold(src=diff_frame, thresh=20, maxval=255, type=cv2.THRESH_BINARY)
5) *Step 5:* Recognizing regions and patterning

III. CONCLUSION

A reliable system supported by tools is necessary for large-scale video surveillance of video streams. A fault resistant and distributed system for video stream analytics can be created using technology like OpenCV, Kafka. A streaming server collector component that takes video streams from various sources and distributes them to a data stream buffer component was created using OpenCV and Kafka. Kafka acts as the component of the data stream buffer that offers long-term archival of streaming data. OpenCV and Spark's Structured Streaming are used in the development of the streaming server processor component. This component analyses streaming data that it receives from the real - time stream buffer using face detection and motion detection

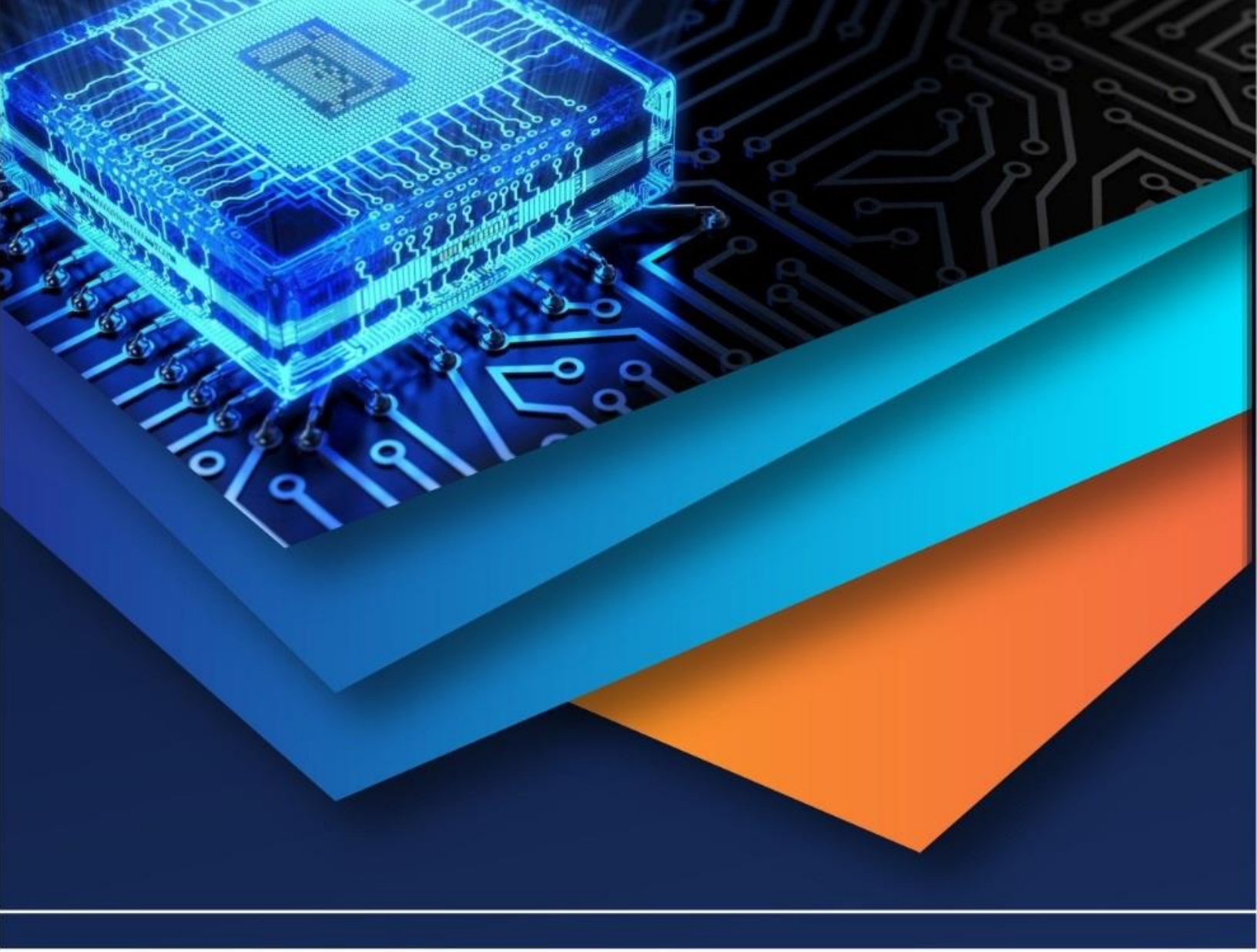
REFERENCES

[1] Kandhalu et al., “Real-Time Video Surveillance over IEEE 802.11 Mesh Networks,” Proc. IEEE Real-Time and Embedded Technology and Applications Symp. (RTAS 09), 2009, pp. 205–214

[2] Grenz et al., “CamInSens-Demonstration of a Distributed Smart Camera System for in-situ Threat Detection,” Proc. 6th Int’l Conf. Distributed Smart Cameras (ICDSC 12), 2018

[3] Real Time Video Analytics for Object Detection and Face Identification using Deep Learning, Vellore Institute of Technology, Chennai Chennai, India Chennai, India International Journal of Engineering Research & Technology (IJERT) ISSN: 2278-0181 IJERTV8IS050298 Published by : www.ijert.org Vol. 8 Issue 05, May-2019.

[4] Wiley STM / Editor Buyya, Srirama: Smart Surveillance Video Stream Processing at the Edge 1 Chapter 13 Smart Surveillance Video Stream Processing at the Edge for Real-Time Human Objects Tracking Seyed Yahya Nikouei, Ronghua Xu, Yu Chen



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY