WWW.IJRPR.COM

**International Journal of Research Publication and Reviews**

Journal homepage: www.ijrpr.com ISSN 2582-7421

# Implementation Paper on Smart Surveillance Via Face Recognition and Cloud Computing

***Monika Parihar*[*1], *Pallavi Urkude*[*2], *Punam Meshram*[*3], *Prof. Abhijeet Thakare*[*4]**

[*1,2,3]Author, Department of Computer Engineering, SRPCE, Nagpur, Maharashtra, India
[*4]Professor, Department of Computer Engineering, SRPCE, Nagpur, Maharashtra, India

**ABSTRACT**

Face recognition has become a major important research area nowadays. One of the important applications in the smart surveillance system is to identify unknown suspicious persons. Instead of manually and tediously monitoring the cameras continuously, this system can be used to identify and recognize suspected individuals and consequently send a warning message on the occurrence of such recognition. The law prosecution uses a software for crime fighting tool using facial recognition .The mobile device usage and the growth of the mobile application has been increased by face recognition .There are many challenges in mobile, laptop and computer device which are facing in problem resources like computing power, limited bandwidth ,battery life and storage so for this, the web face detection and recognition system has been developed. Face recognition is a non-intrusive method, and facial characteristics are probably the most common biometrics features used by humans to identify others.

The process is broken into two steps face detection and face recognition to identify people. For the first step, the system tracks and selects the faces of the detected persons. An efficient recognition algorithm is then used to recognize detected faces with a known database. This system can be implemented at different restricted areas, such as at the office or house of a suspicious person or at the entrance of a sensitive installation. The system works almost perfectly under reasonable lighting conditions and image depths.

*Keywords — Face detection and recognition, AWS, CNN, CDN.*

## I. INTRODUCTION

The ability to detect faces in images and video has become an increasingly important application of machine learning in recent years. There are many use cases for face detection, including security, marketing, and photography. Face detection has become more accessible to developers thanks to libraries like face-api.js, a JavaScript library that allows for real-time face detection in web browsers. In this project, we will be using face-api.js to detect faces in a web browser and host it on AWS.

Face Detection: An algorithm searches for faces in each input, such as video or images, and then cleans the faces with various filters before further processing.

Face Recognition: The Face Detection algorithm's output is fed into the Face Recognition algorithm as input. As a result, the entire process is prone to errors, as in human nature.

Continuously monitoring the surveillance videos requires the outmost attention and the individuals must work expeditiously so that the necessary actions are taken in a timely manner and the delinquent individuals are apprehended. Even the slightest delay could aid the escape of a suspected individual. Therefore, human intervention hinders the overall performance of the surveillance system. As a result, there is a strong need to automate the proposed smart surveillance system to record critical events, detect and even recognize the person.

The suggested system's major goal is to boost intelligence in video surveillance and thereby minimize reliance on human assistance in surveillance-related tasks. By diminishing human interactions in the system, the errors and delays associated with the mentioned interactions can be precluded and thus the performance of the overall system is improved. Additionally, the awareness of security personnel can also be enhanced. The surveillance system is implemented by first detecting faces from the input camera using face js framework to detect. Face recognition is used for the most important types of biometry. These rules help us to see the face quickly with the notification of that person's name in the email and cell phone of the manager. It will also issue a warning if there is an unknown or suspicious person. This face detection system is an excellent tool for preventing terrorist activities.

## II. METHODOLOGY

Loading the required dependencies**:** Before using the Face-api.js library, you must first load the required dependencies, such as the TensorFlow.js library and the pre-trained models for face detection, recognition, and tracking. Preprocessing the image The input image is preprocessed by converting it to a tensor and resizing it to a fixed size that is suitable for the face detection model.

Performing face detection: The face detection model is applied to the preprocessed image to detect the location of one or more faces in the image. The output of this step is a list of bounding boxes, each of which corresponds to a detected face.

Extracting face landmarks**:** The face landmarks model is applied to each detected face to extract a set of key facial features, such as the location of the eyes, nose, and mouth. These landmarks can be used for tasks such as face recognition and emotion detection.

Drawing bounding boxes and landmarks: The output of the face detection and landmarks models is used to draw bounding boxes around each detected face, and to overlay the extracted landmarks on top of each face.

- Optional: Face recognition or emotion detection: If desired, additional models can be applied to the detected faces to perform tasks such as face recognition or emotion detection. For example, the Face-api.js library includes pre-trained models for face recognition that can be used to identify individuals in the input image.

The methodology for implementing the face detection project using JavaScript and hosted on AWS. We will cover the data collection and preprocessing, fine-tuning of the face detection model, integration of the face-api.js library into the web application, and deployment on AWS.

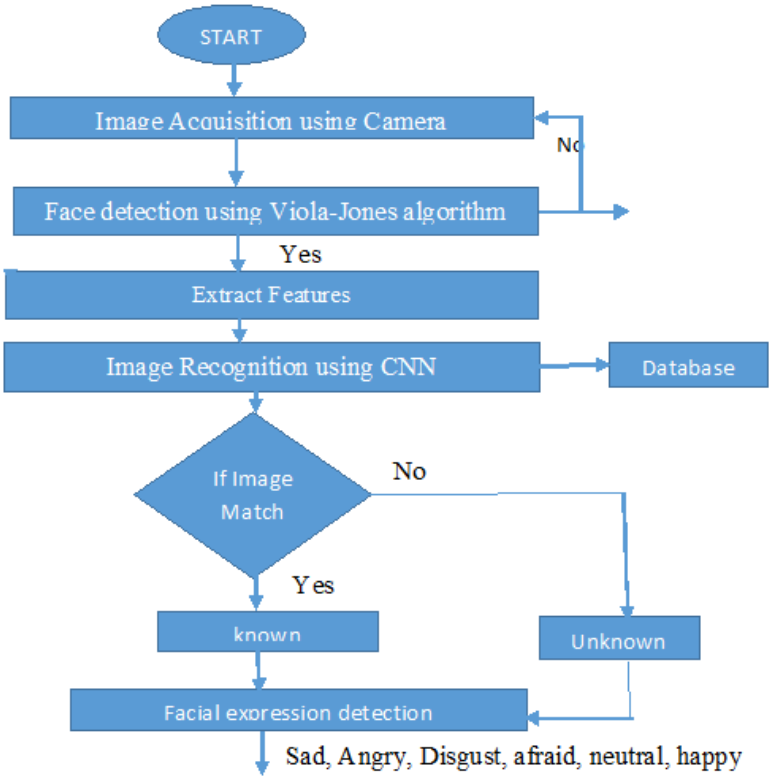



Fig a. Flowchart of face detection and recognition

## III. TECHNOLOGIES

TensorFlow.js: TensorFlow.js is an open-source JavaScript library developed by Google for building and training machine learning models in the browser and on Node.js. Face-api.js uses TensorFlow.js to run the pre-trained face detection, recognition, and tracking models.

HTML5 Canvas: HTML5 Canvas is a drawing API that allows for dynamic, scripted rendering of 2D shapes and bitmap images. Face-api.js uses the Canvas API to draw bounding boxes around detected faces and to display the results of facial recognition.

Node.js: Node.js is an open-source JavaScript runtime built on Chrome's V8 JavaScript engine. Face-api.js can be used on Node.js to perform face detection, recognition, and tracking on server-side images and videos.

Web Workers: Web Workers are a browser API that allows for running JavaScript code in background threads, without blocking the main thread. Face-api.js uses Web Workers to parallelize the computation of face detection and recognition, resulting in faster processing times.

WebGL: WebGL is a JavaScript API for rendering interactive 3D and 2D graphics within any compatible web browser without the use of plug-ins. Face-api.js uses WebGL to accelerate the rendering of facial landmarks, resulting in smoother tracking of facial movements.

Node Canvas**:** Node Canvas is a library that allows for using the HTML5 Canvas API in Node.js. Face-api.js uses Node Canvas to draw bounding boxes around detected faces and to display the results of facial recognition in server-side applications.

JSDoc**:** JSDoc is a documentation generator for JavaScript code. Face-api.js uses JSDoc to generate API documentation for the library, making it easier for developers to understand and use the library. Overall, the use of these technologies allows Face-api.js to provide accurate, real-time face detection, recognition, and tracking capabilities in the browser and on Node.js, making it a powerful tool for building face-related applications.

## IV. LITERATURE SURVEY

[1], Sophisticated tree-based model for face recognition in hazy seasons. The inside and outside pattern had reduced the cost of the PC without lowering the image quality. The three tests showed that the example achieved a precision of 98.65%, 99.19%, and 95.84%. Using databases, this method achieved an overall accuracy of 99%.

[2], A face identification method using deep learning. The Raspberry Pi was used as the main controller to recognize the face. This system always provides a safe and secure lifestyle. Face authentication was tested by two methods, which are testing image and actual time for the system accuracy. It took a long time to work on the image taken.

[3], The face obstacle recognition framework for the security, which was used to find the criminal activities. This method finds the faces of all age groups. Still there were various types of obstacles in faces this method reaches 98.89% accuracy rate. Image identification, image tracking, image authentication are the three stages which are used in this method. Facial acquisition illumination condition can affect the recognition accuracy of this method.

[4], The bank safety locker system using face recognition method. This paper gives a method for modifying the pattern thinning approach to enhance the capabilities of palm vein detection systems. Canny edge detector and conquer techniques are the two modules used for face recognition. A person's palm vein pattern was identified by the Palm Vein Recognition, and it matches the data which is stored in a database for identification.

[5], Face recognition technology for home security. In this system, passive infrared and ultrasonic sensors swere used to connect the web camera to the Raspberry Pi. The user can easily see the activity happening in a house by the simple android application. The system was activated also in power failure with the help of battery. It does not show exact face recognition and to expand the system precision, high expensive sensors are used.

[6], IOT-based face recognition system for home security. The face was recognized using a local binary histogram. Raspberry pi 3 microcontroller was used to build the system. The picture which was used for authentication purposes is 480*640 pixels. In normal lightning the clarity which was found 90% and in low lightning the clarity which was found 80%. If the home's owners don't have a working internet connection, there is not any option to send an SMS to them, which is one of the failures in this system.

## V. PROPOSED SOLUTION

Set up a development environment: To start, you'll need to set up a development environment with Node.js and a text editor or IDE. You'll also need to install the face-api.js library and any other dependencies you plan to use.

-Collect face data: To train the recognition model, you'll need to collect a dataset of images of people's faces. This dataset should include a variety of lighting conditions, facial expressions, and angles to ensure that the model can recognize faces in a wide range of scenarios.

-Preprocess the face data: Before training the recognition model, you'll need to preprocess the face data to ensure that it is in the correct format. This may involve resizing images, cropping faces, and converting the data to a format that can be used by the face-api.js library.

-Train the recognition model: Once the face data has been preprocessed, you can train the recognition model using the face-api.js library. This will involve selecting a pre-trained model, fine-tuning it on your face data, and evaluating the model's performance to ensure that it can accurately recognize faces.

-Build a face detection and recognition app: With the recognition model trained, you can build an app that uses the face-api.js library to detect and recognize faces in real time. This app may include features such as face tracking, face recognition, and emotion detection

-Test and refine the app: Once the app is built, you'll need to test it thoroughly to ensure that it is reliable and accurate. You may need to adjust the model or app code to improve performance, and you should continue to collect data and retrain the model to improve accuracy over time.

-Deploy the app: Once the app is tested and refined, you can deploy it to a web server or cloud platform so that others can use it. You may also want to add additional features such as user authentication, data storage, and analytics to make the app more robust and user-friendly.

This will require a significant amount of data collection, pre-processing, and model training, as well as proficiency in JavaScript and the face-api.js library. However, with careful planning and execution, you can build a powerful face detection and recognition app that can be used in a variety of applications.

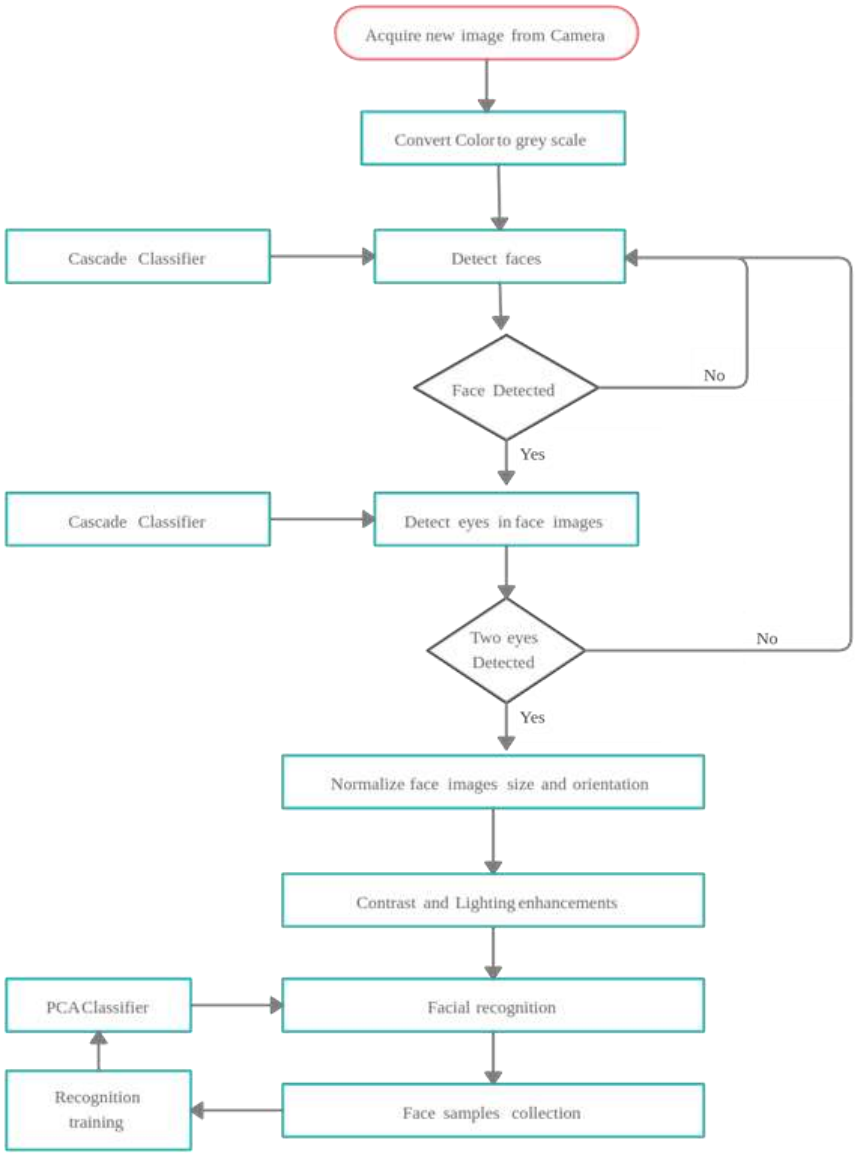



Fig b. Flowchart of Face Recognisation

## VI. IMPLEMENTATION

In this article, we will learn about face detection (Age/ Gender/Face position /mood) using face-api.js and the nearby object detection(person/phone etc) using coco-ssd model on the web browser.

Face-api.js is a java script module, built on top tensorflow.js core, which implements several CNNs(CONVOLUTION NEURAL NETWORK) to solve face detection face recognition and face landmark detection, optimized for web and for mobile device.

Install the face-api.js library: First, you'll need to install the face-api.js library and its dependencies using either NPM or Yarn. You can find detailed installation instructions on the project's GitHub page.

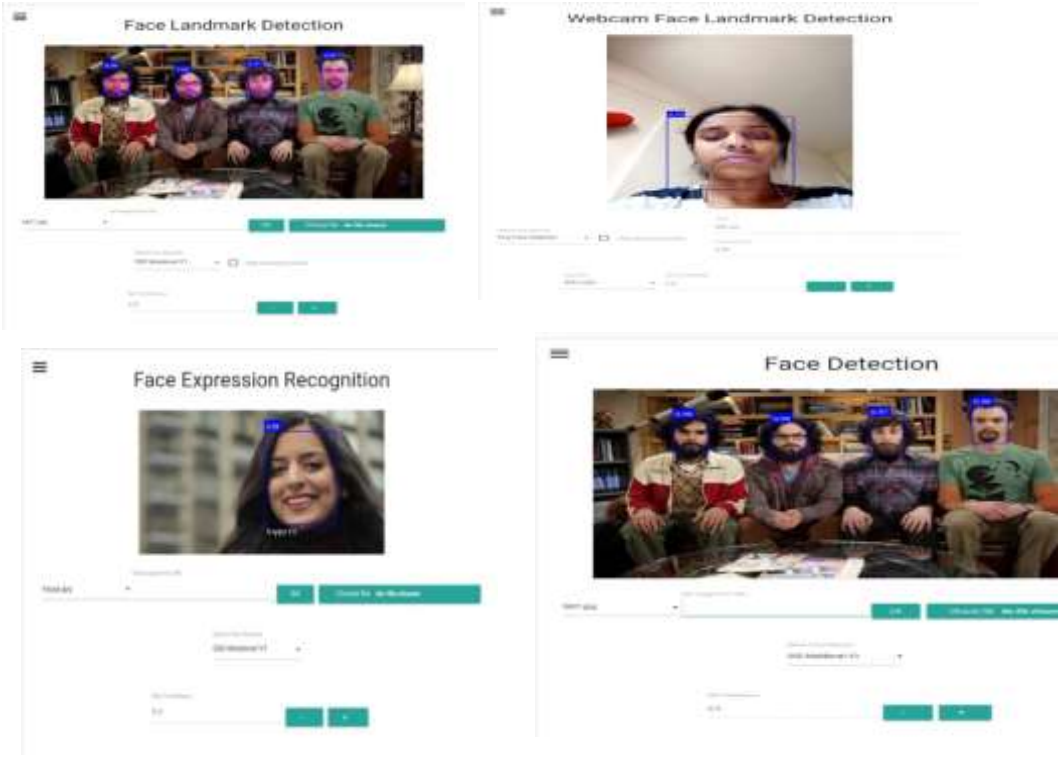
Load the required models**:** Before you can perform face detection and recognition, you'll need to load the required pre-trained models into memory. The face-api.js library provides several models for detecting faces, recognizing faces, and performing other related tasks.

Load an image or video stream**:** Once the models are loaded, you can load an image or video stream that contains one or more faces. You can load images using the HTML5 canvas or an image tag, or you can load video streams using the get User Media API.

Perform face detection**:** Once the image or video stream is loaded, you can use the face-api.js library to perform face detection on the image or video frames. The library provides several methods for detecting faces, including detect All Faces and detect Single Face.

Perform face recognition**:** Once the faces have been detected, you can use the face-api.js library to perform face recognition on the detected faces. The library provides several methods for recognizing faces, including compute Face Descriptor and compute Face Landmarks.

Display the results **:** Finally, you can display the results of the face detection and recognition process on the screen. You can use the HTML5 canvas or an image tag to display the original image with bounding boxes around the detected faces, or you can use the get User Media API to display the video stream with overlays showing the detected faces. Overall, the face-api.js library provides a powerful set of tools for performing face detection and recognition using JavaScript. With a little bit of coding, you can build robust face-related applications that can detect and recognize faces in real time.




## VII. APPLICATION

1. Security companies are using facial recognition to secure their premises.
2. Immigration checkpoints use facial recognition to enforce *smarter* border control.
3. Fleet management companies can use face recognition to secure their vehicles.
4. Ride-sharing companies can use facial recognition to ensure the right passengers are picked up by the right drivers. IoT benefits from facial recognition by allowing enhanced security measures and automatic access control at home.
5. Law Enforcement can use facial recognition technologies as one part of AI-driven surveillance systems. Retailers can use facial recognition to customize offline offerings and to theoretically map online purchasing habits with their online ones.
6. Access Control for Households, private spaces.
7. Face recognition for accessing ATM machines.
8. Automated attendance using Face data.
9. Providing secure access to all devices connected to WSN network
10. Replacement for digital signature by delivery systems
11. Activity monitoring by video analytics
12. Securing transportation by authorized logistics personnel
13. Using Face access instead of keys for unlocking vehicles

## VIII. FUTURE SCOPE

Some of the aspects worth noting for future improvements are as follows-

Liveness Detection System: During the process of development of Face Recognition system, it was observed that the system can be spoofed by using a photo of the owner for access. Hence, a lively detection system was incorporated during the testing of the system. The system can distinguish faces as valid or invalid based on whether the captured face is from a phone or the person himself. However, the disadvantage of the system was that it does not ˘generalize ´ well on all faces. Actual methods also include detection eyes blinking, motion sensing, how pixels change and 3D depth sensing. Depth sensing required cameras with depth sensors for determining if the face is a 2D image or a 3D solid object. Due to lack of suitable hardware and low recall of the method, it was discarded in the final design of the system. Nonetheless, it forms a crucial part in improving the system.

Non-Frontal Face Detection: Only front faces were detected in the final model. The alternative model using Deep Face could allow for non-frontal faces as well. The descriptor of the current model can be combined with it to achieve recognition of non-frontal faces with very low latency.

System Performance: The lower performance of the system can be comphen sated by using dedicated GPUs present in embedded platforms like Nvidia Jetson boards and Google Coral TPU board.

Camera Tampering Detection: It can be created by using image processing and predicting by rate at which frames change.

Indoor Surveillance: Intrusion and access control methods provided above can be also used with indoor surveillance where the cameras are used for monitoring infants in case parents are away and check if pets are safe.

Drone surveillance: The above system can be incorporated in drones for close quarter surveillance operations and identification of people from database.

## IX. CONCLUSION

A system that focuses on face detection and face recognition from a video camera. Implementation simplicity, inexpensive computation, real-time nature and smart acquisition of facial images are some of the features of this system. Face detection is always a challenge, especially face recognition under different lighting conditions, positions and image depths. We found good results when we gathered faces under reasonably varied illumination conditions, positions and depths like those of our database. As the areas under surveillance usually have sufficient illumination condition and cameras can be placed anywhere, our proposed surveillance system can be implemented at offices, educational institutions and different sensitive installations. We may consider more than one camera at different positions and angles. Intelligent techniques can then be used to help combining all the videos from the camera to make the system more efficient, robust and less error-prone to position variations and image depths.

## X. REFERENCES

[1]. Rahul Rawat, Ratnesh Pd Srivastava, “IOT Based Surveillance System Using DNN Model”, ISSN: 2321- 9653, Vol: 9, Issue XII, Dec -2021

[2]. A.D Deshmukh, M.G NakaraniBhuyar, U.B. Shinde, “Face Recognition Using OpenCV Based on IoT for the smart door” SSRN, January -2019.

[3]. C.Muragan, H. Balachander, M. Beston James, “Raspberry Pi Based Smart Surveillance Enhanced with Wi-Fi Technology”, IJET, Vol 7(4.6) .2018, 559-562.

[4]. Raksitha BM, Sandhya V, “IOT Based Real-Time Surveillance Using Raspberry Pi”, ISSN: 2349 – 4689, 2017.

[5]. Rohan Namdeo, Sahil Sharma, Varun Anand, “Smart automated Surveillance Using Raspberry Pi”, ISSN 2277-3878,issue -2, July 2020

[6] K. Jin, X. Xie, F. Wang, X. Han and G. Shi, "Human Identification Recognition in Surveillance Videos," 2019 IEEE International Conference on Multimedia & Expo Workshops (ICMEW), 2019, pp. 162-167.

[7] L. Fu and X. Shao, "Reseach and Implementation of Face Detection, Tracking and Recognition Based on Video," 2020 International Conference on Intelligent Transportation, Big Data & Smart City (ICITBS), 2020, pp. 914-917.

[8] Zafar, U., Ghafoor, M., Zia, T. et al. Face recognition with Bayesian convolutional networks for robust surveillance systems. J Image Video Proc. 2019, 10 (2019).

[9] Sathiyavathi, V., M. Jessey, K. Selvakumar, and L. SaiRamesh. "Smart Surveillance System for Abnormal Activity Detection Using CNN."Advances in Parallel Computing Technologies and Applications 40 (2021): 341.

[10] S. Manna, S. Ghildiyal and K. Bhimani, "Face Recognition from Video using Deep Learning," 2020 5th International Conference on Communication and Electronics Systems (ICCES), 2020, pp. 1101-1106. 392 L. HarikaPalivela et al. / Smart Surveillance System

[11] M. Khan, S. Chakraborty, R. Astya and S. Khepra, "Face Detection and Recognition Using OpenCV," 2019 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS), 2019, pp. 116-119.

[12] A. Bharadwaj K H, Deepak, V. Ghanavanth, H. Bharadwaj R, R. Uma and G. Krishnamurthy, "Smart CCTV Surveillance System for Intrusion Detection with Live Streaming," 2018 3rd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), 2018, pp. 1030-1035.

[13] A. Hampapur, L. Brown, J. Connell, S. Pankanti, A. Senior and Y. Tian, "Smart surveillance: applications, technologies and implications," Fourth International Conference on Information, Communications and Signal Processing, 2003 and the Fourth Pacific Rim Conference on Multimedia. Proceedings of the 2003 Joint, 2003, pp. 1133-1138 vol.2.