Case Study on

Smart Surveillance Video Stream Processing at the Edge for Real-Time Human Objects Tracking

By

Y. PRADEEP REDDY

K. VINAY

P. VENKATA DEEPAK

Regd. No.:

BU21CSEN0600051

BU21CSEN0600070

BU21CSEN0600117

CLOUD-BASED IOT (BATCH NO-1)

CSEN4011

GITAM

DEEMED TO BE UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Gandhi Institute of Technology and Management

(DEEMED TO BE A UNIVERSITY)

BENGALURU, KARNATAKA, INDIA

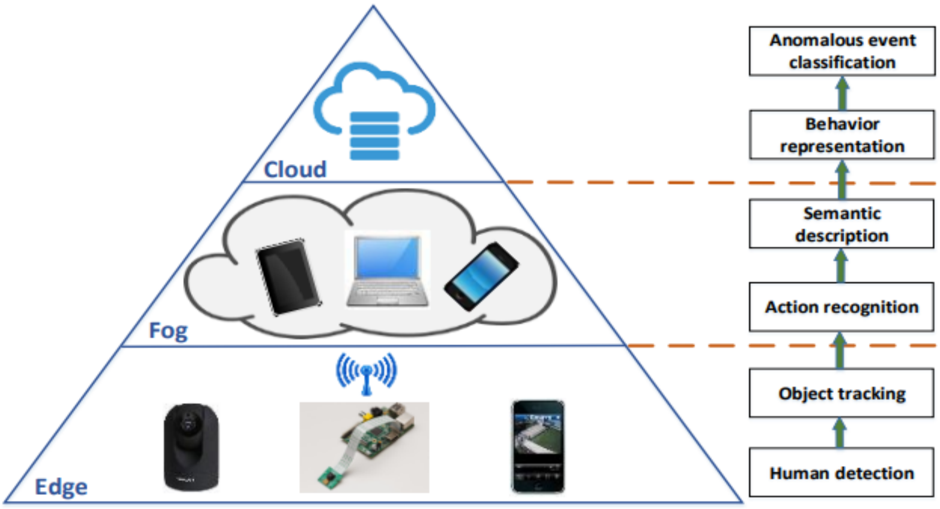
SESSION:2021-2025

CONTENTS

Introduction

Advanced information and communication technologies (ICT) that connect social objects and cyber-physical systems make the idea of "smart cities" possible. It offers valuable services that raise inhabitants' standards of living. Intelligent monitoring is one of the most extensively studied aspects of smart cities. It makes Numerous intriguing applications possible, such as human identification or behavior recognition, crowd flux statistics and congestion analysis, detection of abnormal behaviors, interactive monitoring with multiple cameras, and access control in areas of interest.

However, there are a lot of obstacles that cloud computing-based smart surveillance apps must overcome in real life. While they necessitate real-time object detection and tracking through the processing of video streams gathered from widely dispersed data sources, like networked cameras and intelligent mobile devices, sending the enormous volume of raw frame data to cloud centers not only introduces additional workload to the communication networks but also introduces timing uncertainty. Additionally, because remote data transfer gives attackers more opportunity to explore, it may result in data security and privacy problems. As a result, the surveillance footage feeds are frequently regarded as a precaution for further forensic examination rather than as a preventative measure to stop questionable actions before harm is brought about. As a result, the technologies that are always evolving are vital and sensitive to security.



Human Object Detection

Detecting human objects in a video frame is the process of recognizing individuals. Determining whether a person is there and, if so, where they are in the picture is the aim.

Algorithms are described below:

1. HOG+SVM (Histogram of Oriented Gradients + Support Vector Machine)
2. KCF (Kernelized Correlation Filters)

1. HOG+SVM

HOG+SVM-based human detection algorithm chooses contour as the HOG feature to distinguish human beings from non-human objects given the assumption that people have similar contours even though they have a different appearance of wares.

The combination of Histogram of Oriented Gradients (HOG) and Support Vector Machine (SVM) is a robust approach for object detection, frequently employed in computer vision and image processing. This method excels at recognizing objects based on their shape and appearance by analyzing the local gradients of an image and using a machine learning classifier to differentiate between objects.

Overview of Histogram of Oriented Gradients (HOG)

HOG is a feature extraction method that focuses on the structure and shape of an object in an image by calculating gradient orientations. The key idea is that the shape and contour of objects can be represented through the distribution of intensity gradients or edge directions.

How HOG Works

The HOG descriptor operates by dividing the image into small regions, called cells, and then calculating the gradient of the pixel intensities within these cells. Gradients represent the rate of change of intensity between adjacent pixels and capture information about the edges and contours of objects. The gradient direction and magnitude are computed for each pixel, and a histogram of these orientations is constructed within each cell.

Steps:

- **Gradient Computation:**
    - For each pixel in the image, the gradient in the horizontal (x) and vertical (y) directions is calculated using Sobel filters or other edge-detection techniques.
- **Orientation Binning**:
    - The orientation of the gradient is quantized into bins (e.g., 9 bins

representing different directions from 0 to 180 degrees or 0 to 360 degrees). Each pixel contributes to the corresponding bin based on its orientation and magnitude.

- **Block Normalization**:
  - To improve robustness to variations in lighting and contrast, neighboring cells are grouped into blocks, and the histograms of the cells within a block are normalized. This ensures that the descriptor is invariant to changes in illumination.
- **Descriptor Formation:**
  - The final HOG descriptor is a concatenation of histograms from all the cells, resulting in a high-dimensional feature vector that represents the entire image or a region of interest.

The effectiveness of HOG comes from its ability to capture local shape information while being relatively invariant to small transformations, such as rotations or shifts in the image.

Advantages of HOG

- **Invariance to Illumination**: The use of gradient information instead of raw pixel values makes HOG more resistant to changes in lighting and shadows.
- **Focus on Shape and Structure:** HOG is particularly effective for detecting objects with distinct shapes and edges, such as pedestrians, vehicles, and faces.
- **Efficiency**: HOG descriptors are computationally efficient and can be easily implemented for real-time applications.

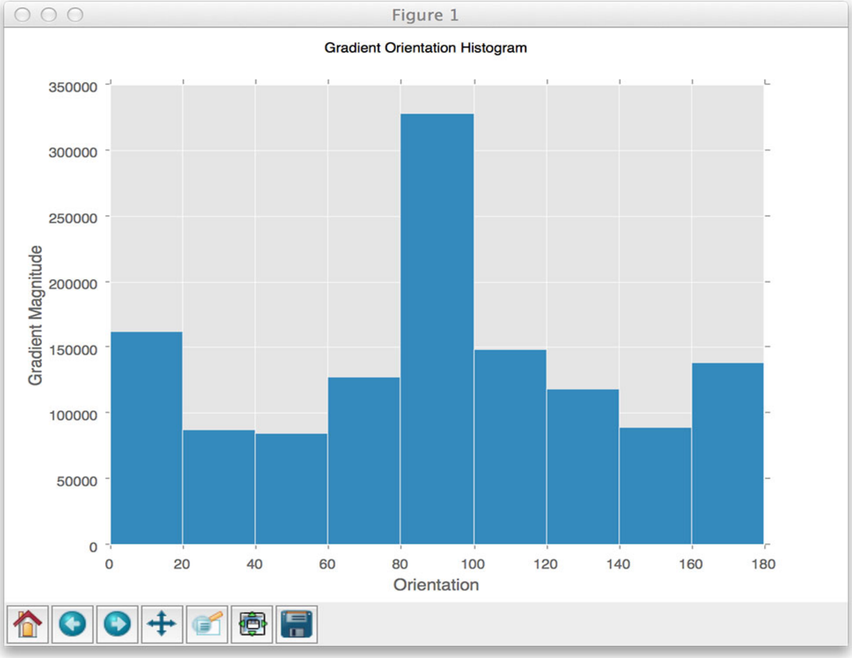


Fig: Histogram of Oriented Gradients

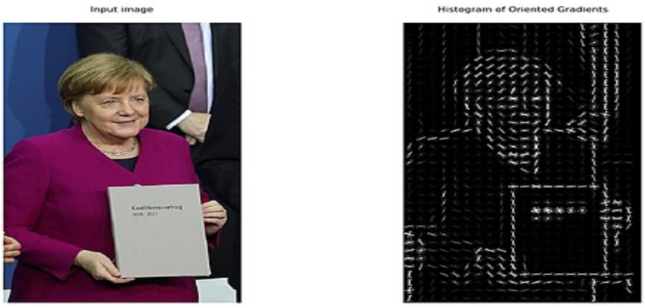


Fig: HOG on an imag

Support Vector Machine (SVM):

Once the HOG features are extracted from an image, they are fed into a Support Vector Machine (SVM) classifier for object detection. SVM is a supervised machine learning algorithm designed to classify data into two categories by finding the optimal boundary or hyperplane that separates the data points.

How SVM Works

The goal of SVM is to find the hyperplane that maximizes the margin between two classes of data points. The margin refers to the distance between the hyperplane and the nearest data points from each class, known as support vectors. By maximizing this margin, SVM ensures that the classifier generalizes well to unseen data.

Key Components of SVM:

- **Hyperplane**:
  - The decision boundary that separates the data points from different classes.
- **Support Vectors:**
  - Data points that lie closest to the hyperplane and influence its position. These are critical for defining the margin.
- **Kernel Function**:
  - In cases where data points are not linearly separable, SVM uses a kernel function to transform the data into a higher-dimensional space where a linear hyperplane can be found. Common kernel functions include the linear, polynomial, and radial basis function (RBF) kernels.

Training the SVM:

The SVM classifier is trained using labeled HOG feature vectors. During training, the SVM learns to classify whether a particular region of an image (represented by its HOG features) contains the object of interest or not. For example, in pedestrian detection, positive samples contain pedestrians, while negative samples contain non-pedestrian regions.

After training, the SVM classifier can be used to scan an image, classify regions, and detect the presence of the object.

Advantages of SVM:

- **Effective for High-Dimensional Data:** SVM performs well with high-dimensional feature vectors like HOG, making it suitable for image data.
- **Robustness to Overfitting**: By maximizing the margin between classes, SVM reduces the risk of overfitting, especially in cases with fewer training samples.
- **Flexibility with Kernels:** Using kernel functions allows SVM to handle non-linear classification problems.

2. Kernelized Correlation Filters

Successful applications of the correlation filter initially inspire KCF in tracking. Compared with more complicated approaches, correlation filters have been proven to be competitive with lower computational power requirements. Object detection-based KCF could be defined as a problem of determining an object's position through template matching that is performed by computing a correlation with a special filter h and subsequent searching of the maximum value on the obtained correlated image c.

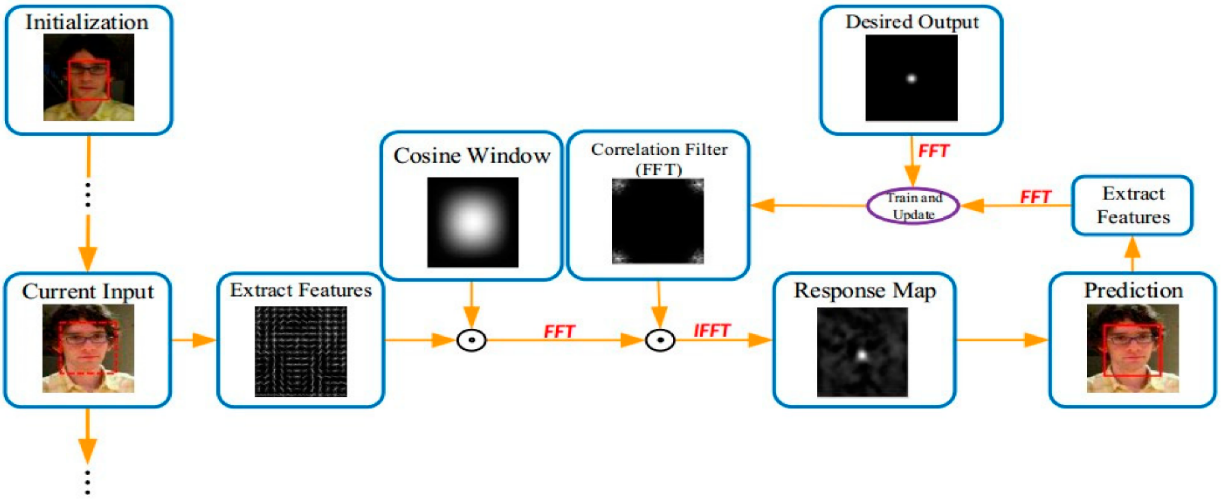


Fig: Kernelized Correlation Filter in Complicated Areas

Correlation filters have been widely used in object tracking due to their computational efficiency and performance. A correlation filter essentially learns a discriminative model of the object in the first frame of a video sequence and then applies that model to subsequent frames to locate the object. Traditional correlation filters work by identifying patterns and features in the image of the object and correlating them with features in subsequent frames. This process is conducted in the frequency domain using the Fourier transform, which speeds up the calculation, making correlation filters suitable for real-time applications.

However, while correlation filters work well in cases where the appearance of the object does not change significantly, they often struggle with more complex, real-world situations. For instance, if an object changes shape, rotates, or experiences significant variations in lighting or background, traditional correlation filters can fail to accurately track the object.

KCF and Fourier Transforms for Efficiency

One of the strengths of KCF is its ability to leverage the computational efficiency of Fast

Fourier Transforms (FFTs). Although KCF operates in a high-dimensional feature space, the use of FFTs ensures that the algorithm remains computationally efficient. In practical terms, this means that KCF can perform tracking at real-time speeds, even when handling complex data.

When an object is first identified in a frame, the algorithm uses the kernelized correlation filter to learn the object's appearance. It calculates the correlation between the object and its surrounding environment in the higher-dimensional space. By operating in the frequency domain, KCF can process these operations much faster than other tracking algorithms that work purely in the spatial domain.

Training and Learning in KCF

In the initial stage of KCF tracking, the object is located in the first frame, and a model is trained to represent the object's appearance. The algorithm builds a response map based on the correlation filter, where the peak of the response map indicates the most likely location of the object. This model is continuously updated in real time as the object moves through the video frames. The key advantage of KCF’s model is that it can adapt to changes in the object’s appearance over time, making the tracking process more resilient to occlusion, changes in scale, and other challenges.

To improve the tracking accuracy, KCF can be combined with HOG (Histogram of Oriented Gradients) features, which are effective in capturing edge and gradient information about an object. These features, when combined with the kernelized correlation filter, make KCF more robust to various appearance changes. The HOG features allow the algorithm to focus on the object’s shape and structure, ignoring irrelevant information from the background.

Strengths and Limitations

KCF’s blend of computational efficiency and robustness to changes in object appearance has made it one of the most popular tracking methods in recent years. Its ability to operate in real-time makes it ideal for applications in surveillance, autonomous vehicles, and robotics, where fast and accurate object tracking is crucial.

In surveillance systems, for example, KCF can track individuals or objects as they move through crowded or complex environments. The algorithm's ability to handle occlusions and variations in appearance ensures that it can maintain accurate tracking even when objects are partially obscured by other elements in the scene. Similarly, in autonomous vehicles, KCF can track objects such as pedestrians or other vehicles in real-time, allowing the system to react quickly to changes in the environment.

Another important application of KCF is in human-computer interaction. For example, KCF can be used in gesture recognition systems, where the algorithm tracks the movement of a person’s hands or body to interpret gestures. The fast and reliable tracking provided by KCF allows these systems to respond quickly and accurately to user inputs.

Object Tracking

Object tracking is a computer vision task that involves monitoring the movement of a specific object across a sequence of frames in a video or real-time environment. The goal is to identify the object in the initial frame and then track its position, scale, and orientation as it moves through subsequent frames.

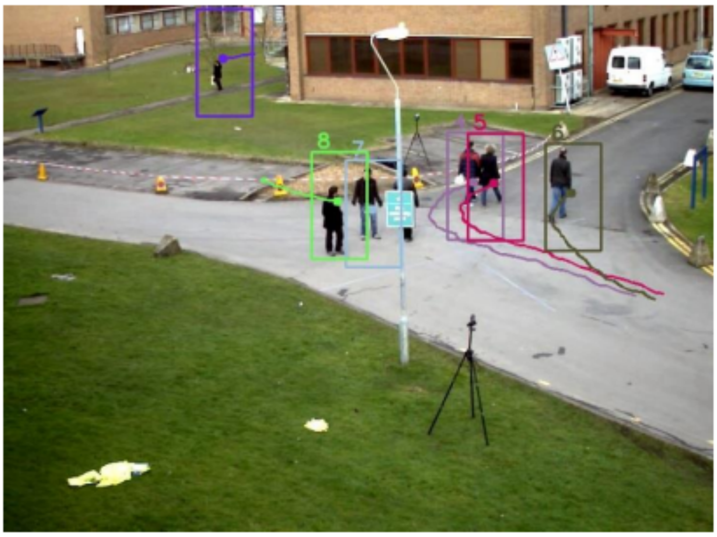


Fig: An example of multi-object tracking

Fig shows an example of the multiobject tracking results. Multi-tracker object queue is designed to manage the tracker's lifetime. Once the human object detection processing is done, the tracker filter compares the detected human and the multi-tracker object queue to rule out the duplicated trackers. Then those newly detected human objects are initialized as KCF trackers and appended to the multi-tracker object queue. During execution time, each tracker runs the KCF tracking algorithm independently on target region by processing the video stream frame by frame until the object phases out, or it loses the object in the scenario

Key Concepts of Object Tracking:

- **Initialization**: The object to be tracked is identified, often by marking its position in the first frame.
- **Tracking**: The algorithm follows the object's movement in real time, adjusting for changes in position, size, and other factors like occlusion.
- **Updating**: The object's appearance may change due to lighting, angle, or perspective, so the tracker continually updates its model of the object to improve accuracy.

Object tracker phase in & out:

The boundary region is defined to handle scenarios which moving objects step in or out of the frame. All detected human objects within the boundary region are considered as tracking targets. They will be added to the multi-tracker queue. In step-out scenarios, those tracked objects that are moving out of the boundary region will be deleted and the corresponding trackers become inactive status. After each frame is processed, those inactive trackers will be removed from the multi-tracker object queue such that the computing resources are relieved for future tasks. The movement history is exported to the tracking history log for further analysis.

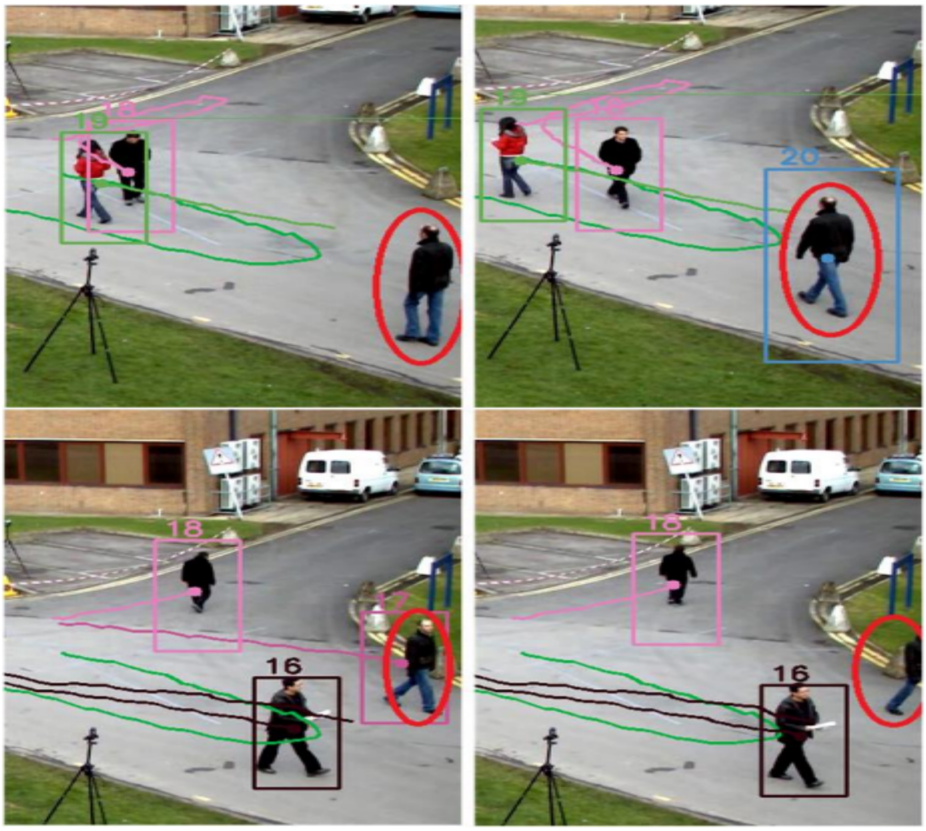


Fig: An example of object tracker phase in & out.

Lightweight Human Detection

Lightweight Human Detection using HOG+SVM and KCF refers to a combination of methods used in computer vision to detect and track humans in a computationally efficient way.

1. **HOG+SVM (Histogram of Oriented Gradients + Support Vector Machine)**:
   - HOG is a feature descriptor that captures the shape and appearance of objects by counting occurrences of gradient orientation in localized image regions.
   - SVM is a machine learning classifier that can be trained to recognize patterns. In this context, it is used to classify whether a given region contains a human or not based on the HOG features.

- Together, HOG+SVM provides accurate human detection by analyzing the appearance of humans in images or videos.

2. **KCF (Kernelized Correlation Filter)**:
   - KCF is a tracking algorithm used for real-time object tracking. After detecting a human with HOG+SVM, KCF can efficiently track the detected human across subsequent frames.
   - KCF operates by creating a correlation filter that predicts the position of the object (in this case, the human) in the next frame, making it computationally lightweight and fast.

By combining HOG+SVM for detection and KCF for tracking, the system achieves efficient and lightweight human detection and tracking, which is suitable for resource-constrained environments like embedded systems or mobile devices.

Conclusion

Smart surveillance video stream processing at the edge for real-time human object tracking represents a significant advancement in the field of computer vision, especially in the context of smart city applications. By leveraging cloud-based IoT technologies, this approach addresses challenges like real-time monitoring, security concerns, and data processing efficiency.

The combination of powerful algorithms such as HOG+SVM and KCF is crucial in human object detection and tracking. HOG+SVM excels at identifying humans based on their shape and gradient features, while the SVM classifier effectively differentiates between human and non-human objects. The use of Kernelized Correlation Filters (KCF) further strengthens the system by offering efficient object tracking in real-time through correlation-based methods that are both computationally light and robust in complex environments.

One of the core innovations in this method is its ability to process video streams at the edge, which reduces the need to transmit large amounts of raw data to cloud servers. This not only minimizes network latency but also improves security and privacy by reducing the exposure of sensitive data.

Moreover, lightweight human detection algorithms like HOG+SVM paired with KCF tracking make it feasible to implement such surveillance systems in resource-constrained environments, like mobile devices and embedded systems. By efficiently detecting and tracking human objects in real-time, this system has the potential to improve safety and security in various smart city applications, including crowd monitoring, access control, and anomaly detection.