SSA: Smart Surveillance Assistant For Mobile Devices

by

Hao Kuang

Thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
In partial fulfillment of the requirements
For the Master of Computer Science

School of Electrical Engineering and Computer Science
Faculty of Engineering
University of Ottawa

Abstract

Over the past few years, the capability of smart devices has grown incessantly, and is showing no sign of slowing down. Along with the decreasing cost of manufacturing high definition surveillance cameras, this has led to the increased ease and convenience of installing surveillance cameras at or in private places. Various applications focus on home surveillance, however simply sending a high definition image does not satisfy all of the users needs. On a site that is monitored by a surveillance camera, the high-resolution surveillance camera streams its video to a user's handheld device. Unfortunately, such devices are unable to make use of the high-resolution video due to their limited display size and bandwidth, and the visual range is also a key problem.

In this thesis, we propose a method to assist the mobile operator of the surveillance camera in focusing on sensitive regions of the videos. Our system automatically identifies relevant regions, combined with foreground detection and human body detection, which is referred to as object detection. A sensitivity map that represents those informative regions returned by the detection methods is accumulated over number of frames. It shows the collection of the sensitivity data in the video over a period of time. We then introduce a zoom strategy to ensure that the operator is able to see the fine details in these areas, while maintaining contextual knowledge. Regions of interest are identified using foreground detection as well as body detection. In order to accelerate the processing speed, we propose two optimization methods. The efficacy of the proposed methods is demonstrated through a user study, the results of which show that this approach is more successful than three comparable approaches used to get an understanding of the activities in a surveillance scene while maintaining context.

Acknowledgements

I would like to give my sincerest appreciation to my supervisor **Prof. Abdulmotaleb El Saddik** for his continuous guidance and support not only in academic level but also in my life.

Special Thanks to:

Dr. Mukesh Saini. His invaluable assistance, guidance, and feedback throughout my research.

Dr. Benjamin Guthier. His concentration and insistence on research affect me a lot.

Dr. Shiai Zhu. He is a mentor during my research time, especially in teaching me how to change thinking for coding an excellent program.

I am grateful to all colleges in MCR lab and all my friends for their comments, encouragement and help during this challenging time.

Finally, I would like to express my deepest thanks to my parents: **Guangming Kuang**(my father) ,**Jianmin Jin** (my mother)and my fiancee **(Crystal)Jingjing Chen**. Their inspiration, understanding and support made this thesis possible. This work is dedicated to them.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

A mobile device is a computing device, typically small enough to be handheld, that has a display screen with touch input and/or a lightweight miniature keyboard. A handheld computing device has an operating system (OS), and can run various types of application software, known as apps. Most handheld devices can also be equipped with Wi-Fi, Bluetooth, NFC and GPS capabilities, which can allow connections to the Internet and other devices. Nowadays, the most widely used mobile device is the mobile phone. From 1983 to 2014, worldwide mobile phone subscriptions grew from zero to over 7 billion, penetrating 100% of the global population and reaching the bottom of the economic pyramid [1], especially in smart phones market. Table 1.1 shows that the leading smart phone manufacturers have over 50% of the market share, the rest being divided by the other smart phones brands. Mobile devices have various uses, including gaming, surfing the net, and watching movies. With the growing capabilities of current mobile devices, inclduing wireless capabilities and improved processing power, surveillance video can be monitored via a mobile device any-time anywhere.

There are various companies and apps that focus on displaying surveillance video on

[1] https://software.intel.com/en-us/intel-ipp

Source	Date	Samsung	Apple Inc.	Huawei	Xiaomi	Lenovo	LG	Others
Gartner[2]	Q3 2014	24.4%	12.7%	5.3%	5.2%	5.0%	N/A	47.5%
IDC[3]	Q3 2014	23.7%	11.7%	N/A	5.2%	5.1%	5.0%	49.%
IDC[4]	Q2 2014	24.9%	11.7%	6.7%	N/A	5.2%	4.8%	46.7%
Gartner[5]	Q4 2013	29.5%	17.8%	5.7%	N/A	4.6%	4.5%	37.9%

Table 1.1: Worldwide Smartphones Vendor Market Share

smart devices. CamioCam [2], a company that uses your smartphone or tablet as a surveillance monitor or camera, has released free IOS and Android apps publicly. Figure 1.1 shows a presentation scenario using that app [3]. iCam [4] is an app that turns your computer's Webcam into a surveillance camera, which you can monitor remotely with your smart devices. AirBeam [5] is another app that convert the cameras equipped with iDevices into flexible remote monitoring systems, therefore allowing for the transmission of high definition video with low latency.

Meanwhile, the cameras employed in the current surveillance systems are also capable of recording high definition videos to provide detailed view of the covered area with low cost.

Consider a scenario where remote video surveillance systems use high-resolution cameras to monitor a security-critical area like a parking lot or the entrance to a home. The captured video is streamed to a remote area, where an operator can view it using a handheld device. However, because mobile devices have smaller displays, where the number of pixels per inch (PPI) is much higher than that of the original monitor, it is hard for the operators to see any details when they are in the full view of the video. A report [8] shows that for a typical 1080p phone (5 display, like the Galaxy S5), the eye starts to notice pix-

[2]https://www.camio.com/

[3]http://venturebeat.com/2014/11/19/camiocam-app-turns-any-mobile-device-or-browser-into-a-home-surveillance-camera

[4]http://skjm.com/icam/support.php

[5]http://appologics.com/airbeam/

[8]http://www.phonearena.com/news/Quad-HD-vs-1080p-vs-720p-comparison-heres-whats-the-difference_id55697

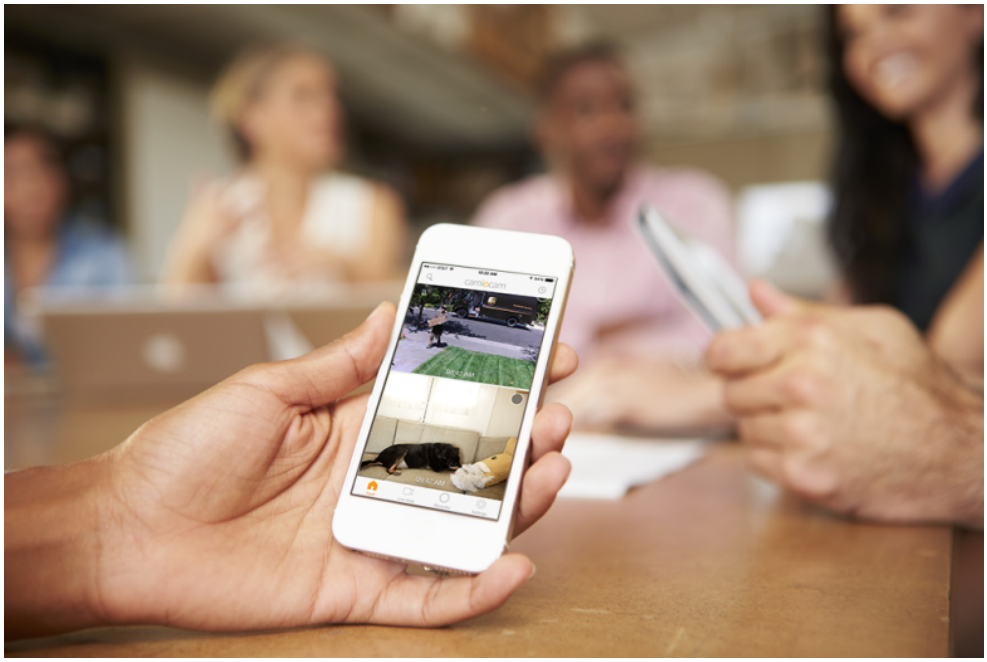


Figure 1.1: Smart phones for displaying surveillance video from CamioCam

elization from 7.8 (19.8cm), which is a close distance between the eyes and the screen, and not made for long-term viewing. Figure 1.2 shows that an average person starts noticing pixelization at around 7.8 inches on a 5-inch 1080p phone. The tester in the picture has 20/20 vision; if you do not, you must be even closer.

While there are some applications that allow users to zoom into the high-resolution video, manual zooming requires that the user continuously zoom into different areas to get the full coverage. The problem with manual zooming is the frequent operations that are required. The user needs to touch the screen, resize the displayed image size, etc. that the problem is that once the user has identified something of interest and wants to focus on it, after continuous screen scaling operations, the target may or may not be in the zoomed area. Furthermore, there is high risk of missing the essential context, as the user can only view a small portion of the video at a time.

Another critical problem is bandwidth. The scenario may not be ideal for transmitting the full resolution video to a bandwidth-limited handheld device. With the development of

[8]http://www.phonearena.com/news/Quad-HD-vs-1080p-vs-720p-comparison-heres-whats-the-difference_id55697

Figure 1.2: Average person starts noticing pixelization at around 7.8 inches in a 5-inch 1080p phone[7]

3G/4G technology, wireless bandwidth is becoming larger and larger, making it possible to develop a greater number of content-rich applications for smart devices. However, except for conditions where free public WIFI is available, the cost of data is not cheap. Research fields such as video coding and video transmission, in order to save data and bandwidth, are also very popular.

1.2 Contribution

In the proposed system, the sensitivity of the regions is determined using two detectors: a human body detector and a background-foreground detector. We chose these detectors because humans and moving objects are potential threats. We fuse the output of these detectors and aggregate them over a period of time to reduce the noise in the detection. The resulting sensitivity is further integrated with user input to obtain the final sensitivity map. Sensitivity means the informative region detected by the proposed system is displayed on one gray image. The most sensitive region of the video is then selected for zooming. To avoid selecting the same region over and over, we penalize the selected region of the video so that the system covers all of the sensitive regions. The system zooms out completely

between transitions to provide enough context to the security operator. The selected region may belong to a moving object, therefore the system can also track the object and adjust the zoomed region so that the output video provides details at a more semantic level.

A complete implementation of the system and its experimental results are provided to validate the work. We found that aggregating the frames over several frames provides the best accuracy of detection. We also analyzed the timing performance of the system and found that the human body detector and the foreground detector are the most time consuming components. To improve the timing accuracy, we performed a series of operations and found that the foreground detection and the human body detection can both be performed on lower resolution frames, without significant compromise to the quality. Finally, the systems performance is evaluated through a user study. The results indicate that the proposed system provides a detailed presentation of the surveillance site in the most effective way.

The main contributions of this work are two-fold:

- Developing a novel video monitoring system custom designed to assist monitoring video through mobile devices.
- Design and implementation of an optimization method for reducing the processing time and an optimization method for increasing the accuracy of detection in ROI detection.

1.3 Publications

- Hao Kuang, Benjamin Guthier, Mukesh Saini and Abdulmotaleb El Saddik. "A Real-Time Smart Assistant for Video Surveillance Through Handheld Devices." Proceedings of the ACM International Conference on Multimedia, pp. 917-920, 2014.
- Hao Kuang, Shiai Zhu and Abdulmotaleb El Saddik. Boosting Prediction of Geolocation for Web Images Through Integrating Multiple Knowledge Sources, ACM International Conference on Multimedia Retrieval, 2015. (Accepted)

1.4 Thesis Organization

The organization of this thesis is as follows:

- Chapter 1 is the introduction. In this chapter, we first discuss the research motivation and issues, and then state the research contributions of this work.
- Chapter 2 is the background knowledge and literature review. The basic knowledge of foreground detection and human body detection are shown as a short survey. Video retargeting and object tracking are also reference works shown in this chapter, after which we introduce a related system in the mobile video surveillance field.
- Chapter 3 is the proposed work. The structure and working flow of the proposed system are presented here.
- Chapter 4 is the implementation. We discuss the software that is used, the library structure of the system, as well as the UML diagram of the system.
- Chapter 5 is the evaluation and results. It provides details on the experiments that allowed us to verify the feasibility and efficiency of our system.
- Chapter 6 is the conclusion and future work. It concludes the thesis and provides suggestions for future work and improvements.

Chapter 2

Background and Related Work

The smart surveillance assistant for mobile devices is related to a number of similar fields of research. This section provides a structured overview of existing works in related fields and connects our work to previous research. The basic knowledge of foreground detection and human body detection are shown as short surveys. Video retargeting and object tracking are also part of the reference works that will be shown in this chapter. Finally, we will introduce related systems that exist in the field of mobile video surveillance.

2.1 Foreground Detection

Foreground detection, also known as background subtraction, plays a very important role in many vision-based applications like video surveillance. It is often used as the first step in image processing, in order to detect various moving objects such as human bodies, hand gestures, and face detection. It therefore needs to be as easy and fast as possible [2]. In our system, foreground detection is responsible for the detection of moving objects in region detection in Section 3.2 to find formative data in the video sequences.

Even though foreground detection has proven its effectiveness and power when subtracting changes in a sequence of images, the problem of separating moving objects in complex scenes is still an issue that has not been completely solved. An effective background sub-

traction should consider and solve several problems such as illumination changes, dynamic backgrounds, camouflaged objects, shadows, bootstrapping, and video noise [8].

In the literature studying background subtraction, there are various existing algorithms [57] [11] [57] [70] [51] [67], but the flow for most of them is very similar. Cheung et al. [11] summarized the difficulties and conceptually proposed a flow structure for background subtraction algorithms. This survey includes various and useful methods, which are discussed under this flow and shown in Figure 2.1.

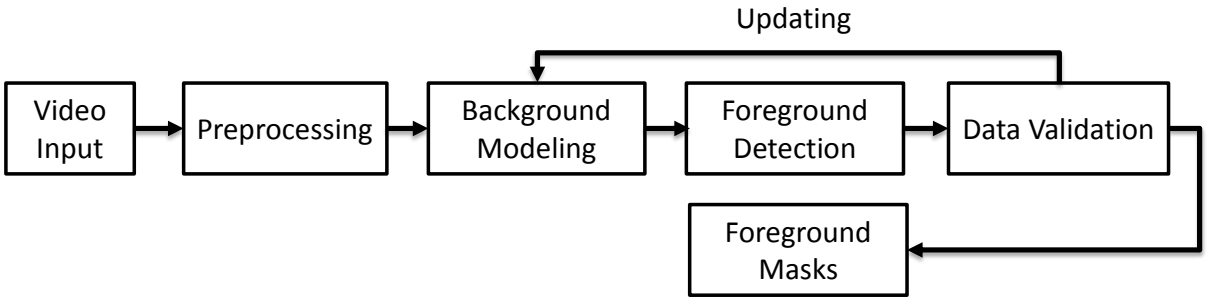


Figure 2.1: Overview flow diagram of background subtraction [11]

Preprocessing is mainly related to two aspects: (1) simple temporal and/or spatial smoothing (2) input data format and aims to accelerate the speed of data processing. Background modeling usually determines how effective the algorithm will be, and in most research papers proves to be very robust against changeable environments. Foreground Detection is a mechanism that splits the foreground pixels from the input image by comparing the input data with the background model. Post-processing is usually applied to the foreground mask, which can range from noise removal operations at the pixel-level to object-level connected components [57].

Due to the importance of background modeling, many different methods have been proposed in the literature; these can commonly be classified into two techniques: (1) recursive techniques (2) non-recursive techniques [11] [57] [70]. A non-recursive technique uses a sliding-window approach to estimate the background, which stores a buffer of previous K video frames. The background image can be estimated based on the variation of the temporal pixels in the buffer. A non-recursive technique is usually highly adaptive and

does not rely on the scene itself too much. Specifically, the number of images in the buffer is fixed and the initial frames are wiped out when the new images come in. However, in order to satisfy the slow-moving objects or temporarily static objects, the size of the buffer needs to be flexible to cope with complex scenes. Unfortunately, it is hard to predict the behavior of moving objects in the next few frames. Frame differencing is a basic common background modeling technique that calculates the difference between the frame at time t and the frame at time t-1 [51]. The median filter is also one of the most commonly used techniques, where the background model is taken from the median index of each pixel in the buffer [10] [18] [16] [31] [47] [82]. Toyama et al. [71] adopt a linear predictive filter to calculate the background model based on the pixel level in the buffer. Another model called the non-parametric model considers the entire history of the frames to form a background estimate, instead of using only one background estimate [24].

In contrast to non-recursive techniques, recursive techniques do not need to create a buffer to approximate background images, but instead maintains a single background model, which is updated by each new video frame. This technique does not require a large fixed buffer. In other words, it reduces the memory requirements and generally improves the speed of computing when compared to non-recursive techniques. Its principal limitation is that once a mistake has been updated in the background model, it will last a certain period of time, causing the foreground mask to be positively affected. The mixture of Gaussians (MOG) model was first used for each background pixel on a traffic surveillance system proposed by Friedman et al [27]. Stauffer and Grimson [67] generalized this work by modeling the recent number of previous images of the color features on each pixel by a mixture of K Gaussians. Other models such as the Approximated median filter [49] and Kalman filter [34] [6] [77] are also popular.

2.2 Human Body Detection

Finding people in individual monocular images is a key problem in computer vision. In recent years, the number of methods used to detect human bodies in monocular images has

increased progressively. The classification of the method used to detect human bodies in various surveys is analogous [53] [41] [25] [22]. Considering the number of various methods, in this section, we only discuss two popular categories: human features including shape and motion, and classifier including the Support Vector Machine (SVM).

Various descriptions of shape information in motion regions, such as points, boxes, silhouettes, and blobs are available to classify moving objects [33] [77] [62] [55] [44]. Haritaoglu et al. [33] propose a read-time visual surveillance system for detecting, tracking and monitoring multiple people in an outside environment. They combine tracking and shape analysis to find people and create their corresponding appearance model. Wren et al. [77] use a multi-class statistical model of color and shape to obtain a 2D representation of people. The automated surveillance system proposed and completed by RetaI et al. [62] is to let only one operator monitor all activities over a complex area. Semantically categorizing detected objects, including humans, is a main part of their work. A Linear Discriminant Analysis (LDA) is used to classify vehicle types and people. Oren et al. [55] propose the idea of using a wavelet template, inspired by the Haar wavelet [48], to define the shape of an object and detect people in static images of cluttered scenes. Lipton [44] proposes a real-time method for extracting and classifying moving objects in a video stream. They presume that humans and vehicles are of more interest to users. A bi-variate approach is employed to classify the data over the sample images.

Generally, the key thread of using motion-based classification is that non-rigid articulated human motion shows a periodic property. The present method can be classified into point corresponding, pixel periodic analyzing, object motion periodic property analyzing, etc. [19] [45] [59] [26] [28] Cutler and Davis [19] split the periodicity detection and analysis into two steps: (1) tracking the objects in the foreground that come from segmentation (2) analyzing a self-similarity metric to detect and characterize the periodicity using a time-frequency analysis method.

Liu and Picard [45] propose a periodicity detection algorithm that consists of (1) object tracking by frame alignment (2) simultaneous detection and periodicity region segmentation, which can be applied to motion classification and recognition. Polana and Nelson [59]

propose a system that identifies periodic activity in discrete gray-level image sequences. Motion detection is done via image differencing, and a simple nearest centroid classification method is used to distinguish motions. They also propose a recognition method under pixel level , where the tracked object is taken from flow based algorithms and a machine algorithm and the nearest centroid classification is applied to recognize several specified motion activities [58]. Fujiyoshi and Lipton [28] adopt a segmentation method to obtain moving objects from a static camera, after which the boundaries of the objects are extracted. A "star" skeleton is a concept they propose, which represents the component parts of a target with internal motion. It is used to analyze the motions that only need a small number of pixels. A shape-based method is generally simpler than a motion-based method as it does not need specific environments and devices.

Discriminative classification techniques aim to find an optimized decision boundary to distinguish pattern classes in a featured space. Utsumi and Tetsutani [72] propose a human body detection method based on a distance map. The distance map is made by $M*N$ small blocks. Then, for each element in a distance, the $MN*MN$ matrix represents the distance between the color distribution in each block. Based on this map, a statistical model is built for each object type in a database. Finally, the Mahalanobis distance is used to find the similar elements. Viola et al. [74] propose two methods, one for static images and the other for videos. The main idea for the static image detector is that it processes object detection on two successive images, and then lets the Adaptive Boosting (AdaBoost) train on human shapes and motion features, to select weak classifiers. The dataset combines both images and videos of human and non-human instances. The dynamic detector is based on the motion and static rectangular features. Support Vector Machine (SVM) [9] [15] is a powerful tool for human detection, and has proven its efficiency for face detection [56] and gender recognition [52]. Human body detection using SVM classified by target appearances, such as edge segments in the image, also performs very well. The work of Sidenbladh focuses on robust detection, which combines SVN and a Radial Basis Function (RBF) kernel to train the human body on optic flow patterns. Dala and Triggs [20] use Histogram of Oriented Gradients (HOG) as the feature for creating classifiers. The key is that an object can be represented by a distribution of intensity gradients or edge directions.

2.3 Video Retargeting

The presented work is most strongly connected to *video retargeting* . Video retargeting typically deals with high resolution input videos that were created for presentation on large screens. Our goal is to change the size or the aspect ratio of such a video to fit a smaller screen size, e.g., the display of a mobile device. As much of the important video content as possible should remain visible in the retargeted version. At the same time, the retargeting process should keep the amount of introduced artifacts, for instance deformed objects, low. Temporal stability is another desirable goal for the retargeted video. The retargeting parameters should remain consistent from one frame to the next in order not to introduce temporally varying artifacts like shakiness or flickering.

A good general survey on video retargeting can be found in [39]. More specifically, the proposed system presents a cropping-based approach. In this family of techniques, a rectangular area is chosen for every frame, such that it contains the highest amount of visual importance for a given size [46] [23] [43] [21]. All pixels outside the rectangle are discarded entirely. The choice of a suitable rectangle is a compromise between losing detail through scaling the video and losing context through cropping. The locations of cropping windows in each frame over time describe a trajectory through the space-time cube of the video. This trajectory is constrained by cinematographic rules for panning, which require a certain amount of smoothness and an adequate amount of zoom [38]. An optimal trajectory may be determined in a number of different ways, e.g., by an exhaustive search through parameter space [46], a shortest path algorithm, a max-flow/min-cut algorithm on a weighted graph [43], or by dynamic programming [21]. If multiple optimal trajectories are found, artificial cuts may be introduced to transition between them. The advantage of cropping-based video retargeting is that the content in the chosen area of interest is preserved faithfully, and few visual artifacts are introduced. A common disadvantage, however, is that important content may be discarded entirely. None of the mentioned algorithms zoom out completely to give an overview of the whole scene.

Importance-based scaling (also called “warping) is another popular group of retargeting approaches [75] [30]. Instead of cutting out important content, the frames are divided into

smaller image areas, which are then scaled down separately according to their respective importance. The goal is to keep the size and shape of important foreground objects nearly identical, while less important background areas are shrunk and may be deformed. This distributes the loss of detail and deformation over the regions of a frame to which the viewer is paying the least amount of attention. Warping approaches are very powerful and are able to produce visually pleasing results for still images. When applying them to videos, maintaining temporal stability during highly dynamic scenes is a major challenge.

Seam Carving is a technique that was originally developed for the retargeting of still images [32], but it has been adapted for the purpose of retargeting videos as well [40]. Seam Carving works by finding a vertical seam of connected pixels that crosses the image from top to bottom. The seam is chosen so that its pixels cover an area with the minimum possible amount of visual importance. Removing the pixels belonging to the seam from every row of the image reduces the width of the image by one. The process of finding and removing seams can be iterated to reduce the width by arbitrary values. The height can be decreased anologously by removing horizontal seams. Due to its tendency to remove pixels from unimportant regions, Seam Carving has been reported to produce excellent results for images with large unstructured areas such as the sky, water or walls. However, problems may occur when the shape of the unstructured area bears importance to the image, or when an image mainly consists of structured backgrounds and straight lines. Also, even though Seam Carving may be applied to each frame of a video individually, maintaining temporal stability and producing satisfactory results for videos remains a major challenge.

Approaches based on both warping or Seam Carving are generally unsuited in a video surveillance scenario. This is mainly due to the modification of the content of the video. Deforming objects in the scene or changing their position and size relative to each other may alter the context and lead to misinformation. Furthermore, the potential for introducing artificial motion through temporal instability is undesirable in a surveillance scenario where object motion is an important cue. In general, all video retargeting approaches, including cropping-based approaches, focus on the aesthetics of the results.

The output videos should be pleasant to watch [78]. Completely removing certain

content may be more desirable than partially cutting off an object, and deformation in low attention background areas may be tolerable. Also, greatly changing the composition of a scene by relocating and resizing objects has been shown to have convincing results [66].

In our surveillance scenario, aesthetics are only a minor goal and may be sacrificed to ensure a more faithful preservation of content and context. The composition of the scene must not be tampered with. It is also imperative that there be no blind spots in the video, where content has been removed permanently. Everything that is contained in the original video should also appear in the retargeted video at some point in time.

2.4 Object Tracking

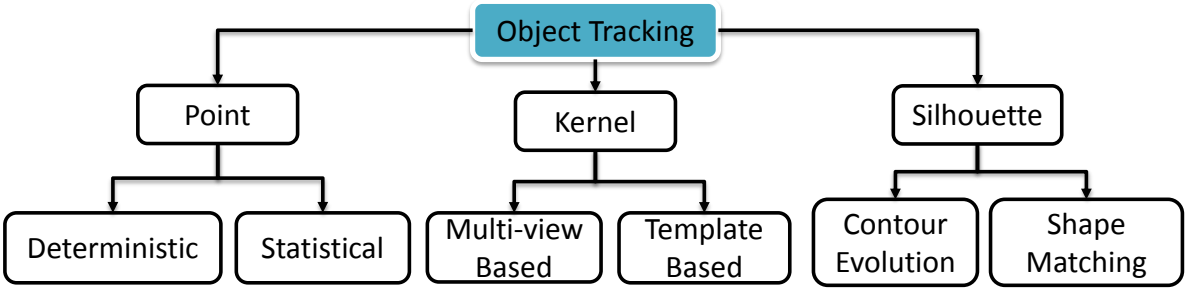


Figure 2.2: Categories of tracking methods [79]

A good general survey on object tracking can be found in [79], where the tracked objects are divided into nine categories: (1) points (2) primitive geometric shapes (3) object silhouettes and contours (4) articulated shape models (5) skeletal models (6) probability density of object appearances (7) templates (8) active appearance models and (9) multi-view appearance models. The feature selection for tracking is also divided into (1) color (2) edges (3) optical flow and (4) texture and summaries some popular methods of object detection [63] [65] [73] [7] [4]. The author proposed a taxonomy of tracking methods, shown in Figure 2.2, which can generally be classified into three categories.

Point tracking is a common method of object tracking, which represents the detected objects as points, in order to create a tracking line over the frames. Salari and Sethi [63] propose an improved tracking algorithm by: (1) listing the correspondence for the detected points (2) adding hypothetical points for the missing tracking points, which is based on the work proposed by [65] which uses a greedy approach to solve the correspondence. Veenman et al. [73] propose a qualitative motion modeling framework combined with various motion models. The key idea is that it uses an optimized strategy for changing models. Broida and Chellappa [7] use the Kalman filter to track multi-points in complex images. Comaniciu et al. [13] use a mean-shift tracker, which maximizes the similarity by comparing the weighted histogram of the object with the current frame to track the target recursively.

Kernel tracking is also a typical computation method performed frame by frame when computing the motion of an object [4]. Support Vector Machine (SVM) is a general classification scheme based on a set of negative and positive training samples and used to find an optimal hyperplane separating two classes. Avidan [1] combined a SVM classifier and an optic-flow-based tracker to track the rear end of vehicles from a video sequence. It maximizes the score generated by SVM classification over the regions of the frame, to estimate the objects position. Isard and Blake [5] use affine motion and spline shape to paint a tracked object state, which is updated by a particle filter. The state variables are used as training data and can be manually obtained during the consecutive frame. Bertalmio et al. [3] propose a Partial Differential Equation (PDE) based method to track objects by deforming curves on consecutive images to get the expected position in the next frame.

Silhouette tracking is performed by estimating the region around an object in each frame [79]. Huttenlocher et al. [36] use an edge-based representation method to perform shape matching. The Hausdorff distance is adopted to establish a correlation surface from the new objects position. Sato and Aggarwal [64] use the Hough transform to generate object tracks for object silhouettes over frames in the velocity space. In contrast to traditional methods of using a histogram, Kang et al. [37] propose a method of histogram generation based on concentric circles, where color histograms and edge are used to establish the object model.

Since visualization, including zooming, is included in our surveillance scenario, tracking the object in each frame efficiently is more important than any other factor, no matter how detailed the representation of the result is.

2.5 Mobile Video Surveillance

In this section we list other systems related to mobile video surveillance.

Zhang et al. [81] propose a distributed architecture for high definition (HD) multi-view video surveillance systems. Multiple intelligent IP-based video surveillance cameras are connected to a local video server, as can be seen in Figure 2.3. Foreground detection, object detection, and trajectory generation are included in this system, which let the final display images indicate moving objects. GPU-based video servers are used for further processes such as multi-view and multiple object tracking. The system theoretically supports multiple devices such as monitors and smart phones, but the paper doesn't show the proof of how to solve the limitations of using these devices, for example, the screen size of a smart phone. Although the results show that the UI interface can support up to nine cameras, the evaluation part only includes two images with their mask image of detection result. Cucchiara and Gualdi [17] propose an architectural overview of mobile video surveillance systems. They mentioned that one of the most significant advantages of using mobile surveillance systems is that they can provide surveillance and remote monitoring wherever and whenever the operator wants, without the need to install a fixed system in a specific location. In the paper, they also made the distinction between the terms mobile and moving, where a mobile module is free to move but operates in a static way, and a moving module can be operated in a dynamic way. A three-features three-dimensional graph shown in 2.4 proposes to differentiate the present works in various regions. The system we propose can be assigned to a category (fixed sources or moving sinks), which belongs to a region (distributed, moving region). One of the features in this region is the requirement for wireless network communication, which introduces limitations in the bandwidths.

Chung proposes a framework for a mobile surveillance service for users who are using

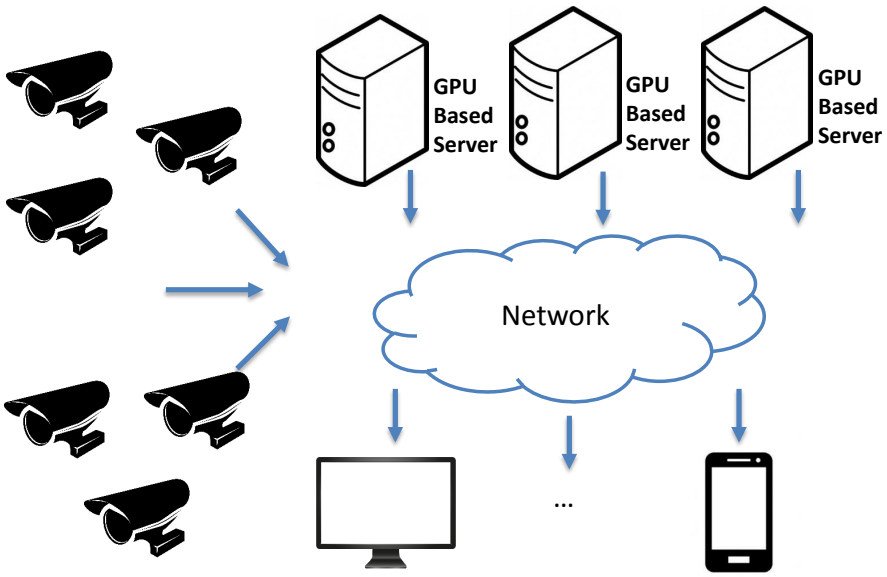


Figure 2.3: Block diagram of High Definition Multi-view Intelligent Video Surveillance System [81]

smart phones [12]. The framework proposed in this thesis is similar to other works, which mainly focus on the communication between multiple video cameras/servers and the mobile client. The system adopts HTTP streaming for a real-time display of images on the smart phones. Meanwhile, the author considered one important feature of using smart phones as surveillance monitors: bandwidth. Three different video transmission models: INTRA, INTER and JPEG were compared in order to get an optimized result. However, even in this research, the limitations surrounding the size of smart phones weren't yet considered.

Wang et al. [76] propose an object-based video recording approach that only records the video clips of abnormal events, which enables the operator to watch only those clips. Moving object detection and a tracking algorithm help the system to keep keyframes. The author skims those frames without informative regions to save the computational resources and transmission bandwidth. The RTSP streaming protocol is adopted to transmit video clips of objects. Figure 2.5 shows the function modules of the system they propose. The test frame size is based on QCIF (176x144 pixels), but nowadays, the trend is to use high

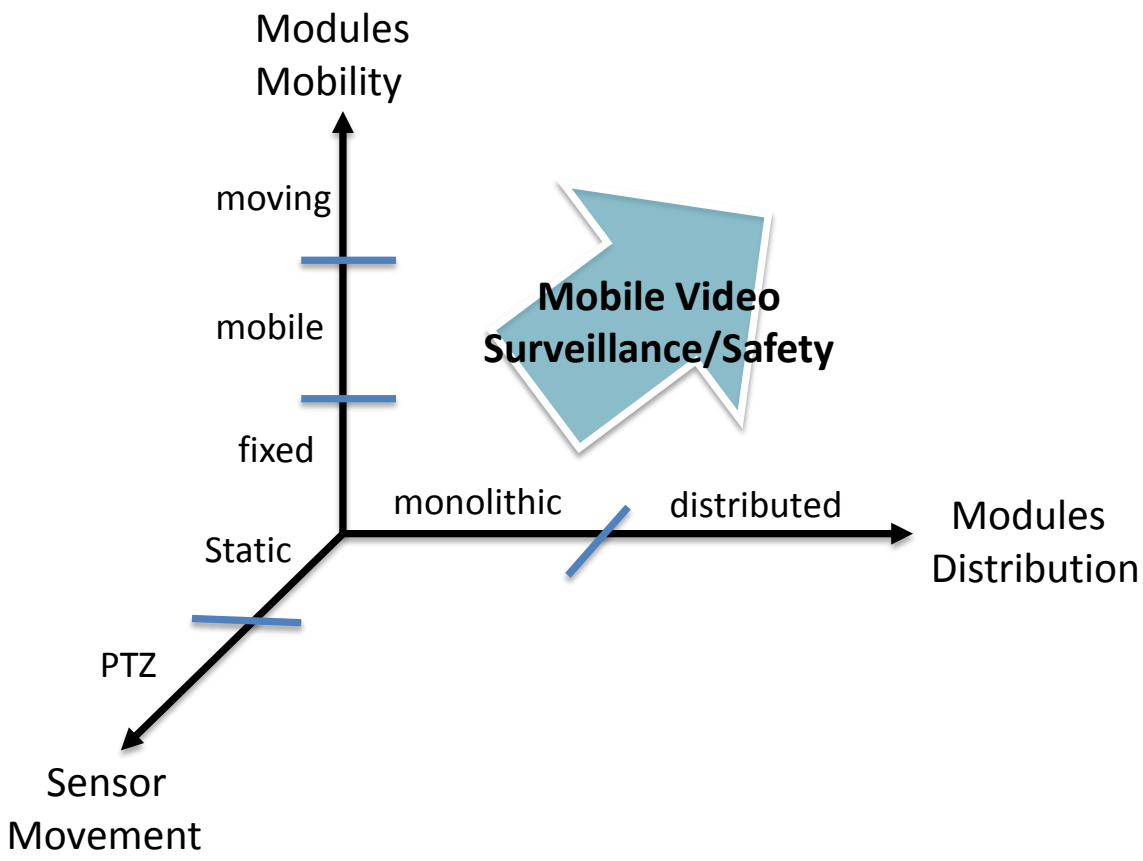


Figure 2.4: The three features of a system for video surveillance or safety

definition surveillance cameras.

Steiger et al. [68] propose a video adaptive encoding scheme that uses object-based adaptation to deliver surveillance video to mobile devices. The method aims to provide the highest perceptual quality for the operator. There are three candidate strategies: coded original, spatial resolution reduction and semantic prefiltering. However, the method doesn't work for a relatively complex scene, for example, if the surveillance scene is some distance away from the camera, the small moving objects can not be distinguished very well.

Rty et al. [61] propose an Area of Interest (AoI) system that consists of the AoI server and the AoI Client. A Video Surveillance Intelligent Platfrom (VSIP) [50] is introduced to handle images from the surveillance camera. Figure 2.6 shows the generic architecture proposed in [50] for the video comprehension system. The AoI system only transmits the

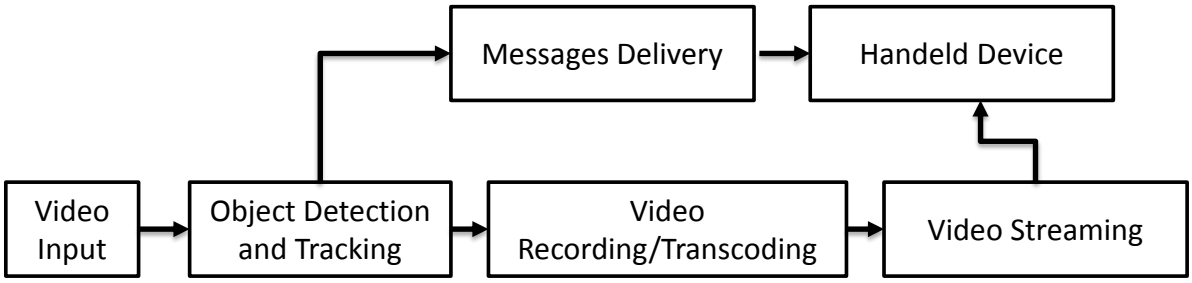


Figure 2.5: Function modules of video surveillance system with intelligent object analysis

secluded tracked object's images to the remote device, which will then be portrayed on the static background. As with previous research, however, the system still does not consider the limited screen size of smart phones, and the operator still does not see the details around the tracked objects.

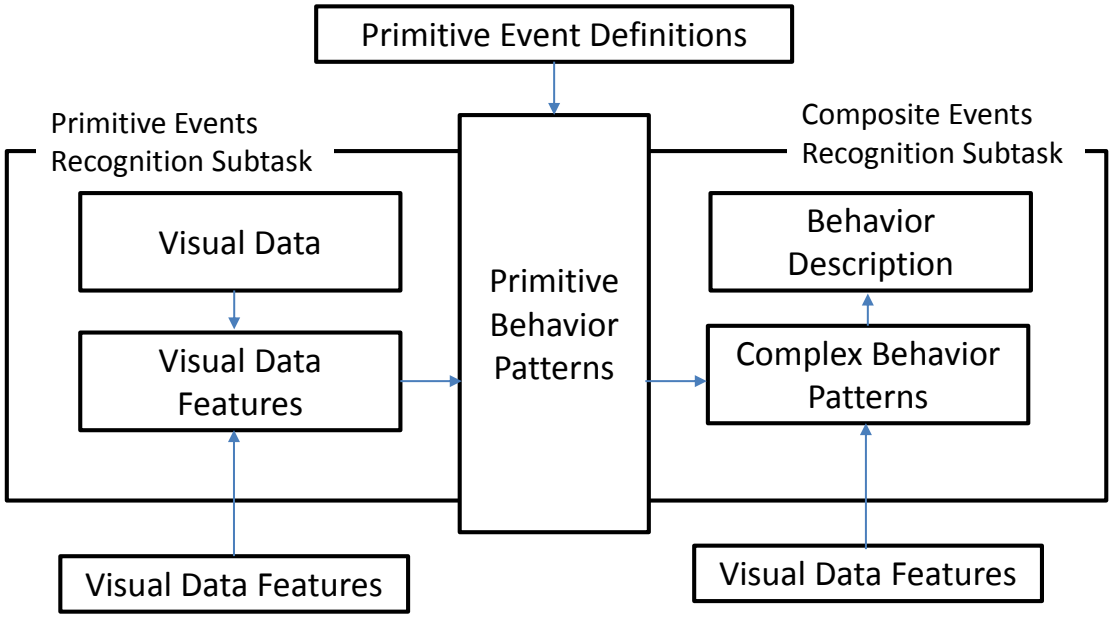


Figure 2.6: A generic architecture of a video comprehension system.

Li et al. [42] propose a mobile surveillance system that is based on the Java development approach. Figure 2.7 shows the layered structure of the PDA Watcher. The end user is able to use PDF to watch the surveillance video. Some new Java technologies including J2ME,

RMI, and JMF are used for multimedia computing and mobile computing. A Watching Transmission Protocol (WTP), which is a HTTP-like protocol, is proposed and used in the communication layer to realize a thin-client goal.

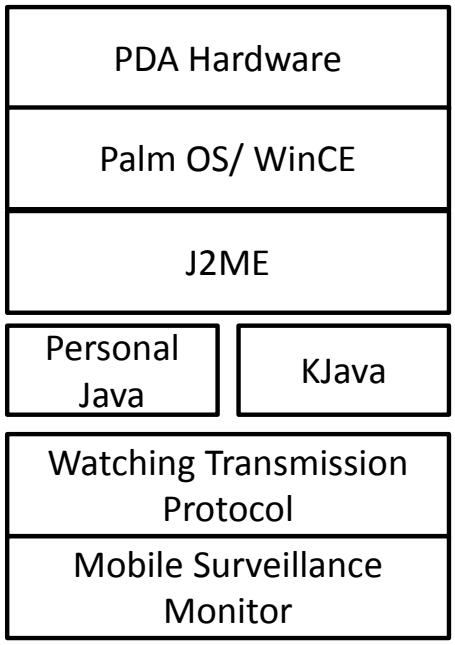


Figure 2.7: The layered structure of the PDA Watcher

Chapter 3

System Design

3.1 SSA: Smart Surveillance Assistant System Overview

The assistant for video surveillance on handheld devices aims to search HD video to find important areas on which to focus. The system will zoom into the area of focus, follow the object for a while, zoom out again, and then the process will start again.

The assistant for video surveillance on handheld devices aims to search HD video to find important areas on which to focus. The system will zoom into the area of focus, follow the object for a while, zoom out again, and then the process will start again.

The system consists of three main components that work together: region detection, ROI selection, and visualization. See Figure 3.1 for an overview of the system and Figure 3.2 shows how it processes the input video streams frame by frame. Once the system gets a frame from a surveillance camera with a high definition (HD) quality image, the frame will be handled by the region detection component over certain frames. Foreground detection and body detection are two main parts of region detection, which function together on the same frame, to detect any moving objects, including human bodies, in the scene. The result from the previous step is accumulated over several images, which aims to reduce the negative effects caused by the size of the detected region, and thus avoid a false detection. In order to accelerate the process and improve the accuracy of the selected region, we

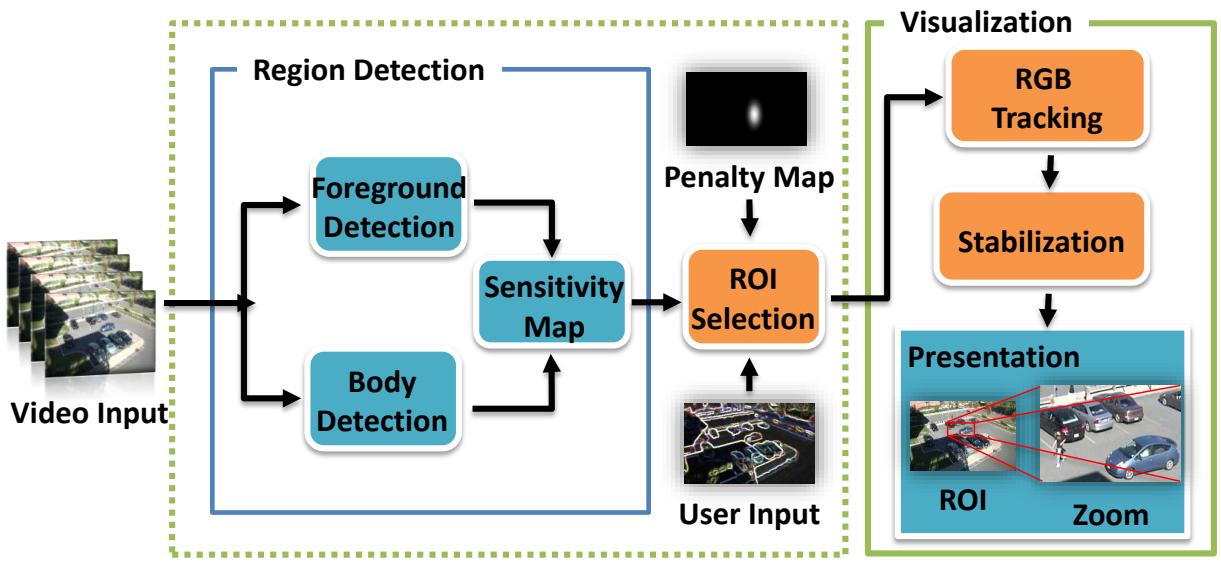


Figure 3.1: Structure of remote video surveillance assistant.

optimize the accumulation time and the resolution methods. The resulting binary images will be accumulated and put together as a gray image, also referred to as a sensitivity map, which shows the informative region in section 3.2.3. Then, based on certain common image processing algorithms, the ROI Selection will choose the most informative region in the sensitivity map. Meanwhile, the penalty map and the user input will also help determine which part of the image is more important. In the penalty mechanism, a 2D Gaussian function ensures that recently selected regions are given less importance in the sensitivity map. User input is a manual input entered by the operator when the surveillance system commences, and aims to remove the regions that are not really related to the main surveillance area. After the most informative region is chosen, it is time for visualization, which includes RGB Tracking, Stabilization and Presentation. RGB Tracking is responsible for tracking the object according to the color histogram of the target, which works in each frame during the whole period of presentation. The first time of starting tracking in the system is after accumulating the sensitivity map over ω. Stabilization is used to remove the shakiness that comes from the RGB Tracking part; it is the key to an improved viewing experience for the operators. The result from the previous steps is a data rectangle including height, weight and top left corner coordinate (x,y), corresponding to the detected region. Afterwards, the presentation part will process this image and display it by zooming

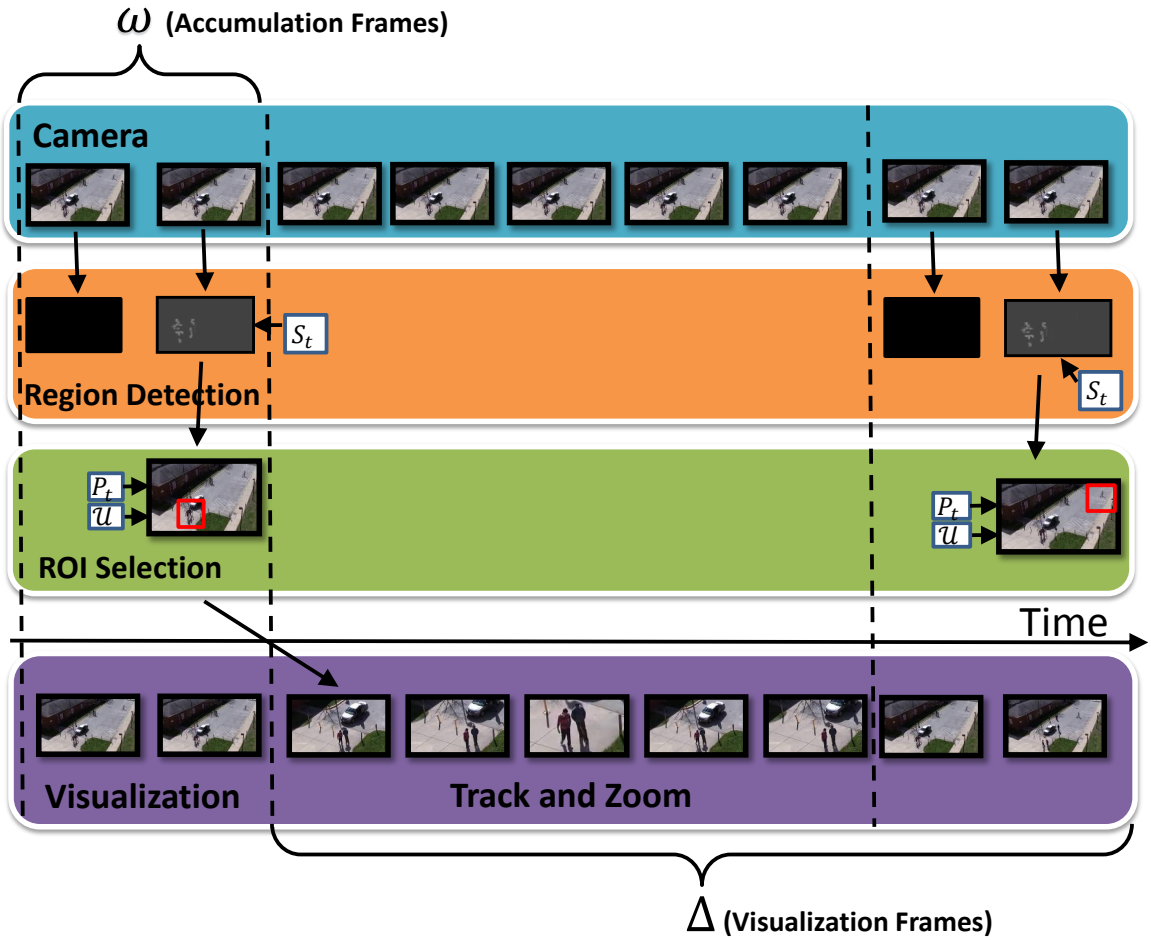


Figure 3.2: Overview of working flow of the system

in on the area on which the operator most likely wants to focus. Finally, the resulting resized frames are sent over the network to the handheld device.

3.2 Region Detection

In a surveillance scenario, objects moving in relation to the static background, i.e. foreground objects, are generally the main focus of attention, and of those moving objects, the human body is of the utmost significance. The sensitivity map is thus computed from the simple accumulation of human body detection and foreground object detection for each frame over ω times.

Region detection includes three steps. Firstly, accumulating and combining the information for the sensitivity map, which aims to reduce the negative effects of the size of the detected region, as well as any false detections in the sensitivity map. Secondly, optimizing the accumulation time of the sensitivity map. We try to find the ideal number of frames over which we should accumulate the two maps, in order to reduce the processing time and increase the detection accuracy. Thirdly, optimizing the resolution of the input image for the detection method, to reduce the resolution as much as possible and therefore reduce the processing time without compromising the accuracy too much.

3.2.1 Foreground Detection

The foreground is defined as the area of the video that is occupied by moving objects. Moving objects are very important in video surveillance. There are various methods to detect motion in a video, for instance temporal differencing, optical flow, and background modeling [35]. The most common of these being background modeling, in which we estimate the background and compare it with a current video sequence [80]. Stauffer and Grimson [67] model each pixel with a Mixture of Gaussians (MOG) and use an online approximation to update the model and make it more adaptive.

In our work, an improved version of this algorithm by Zivkovic et al. [84][85] is used to determine the foreground mask. An important feature of the improved algorithm is

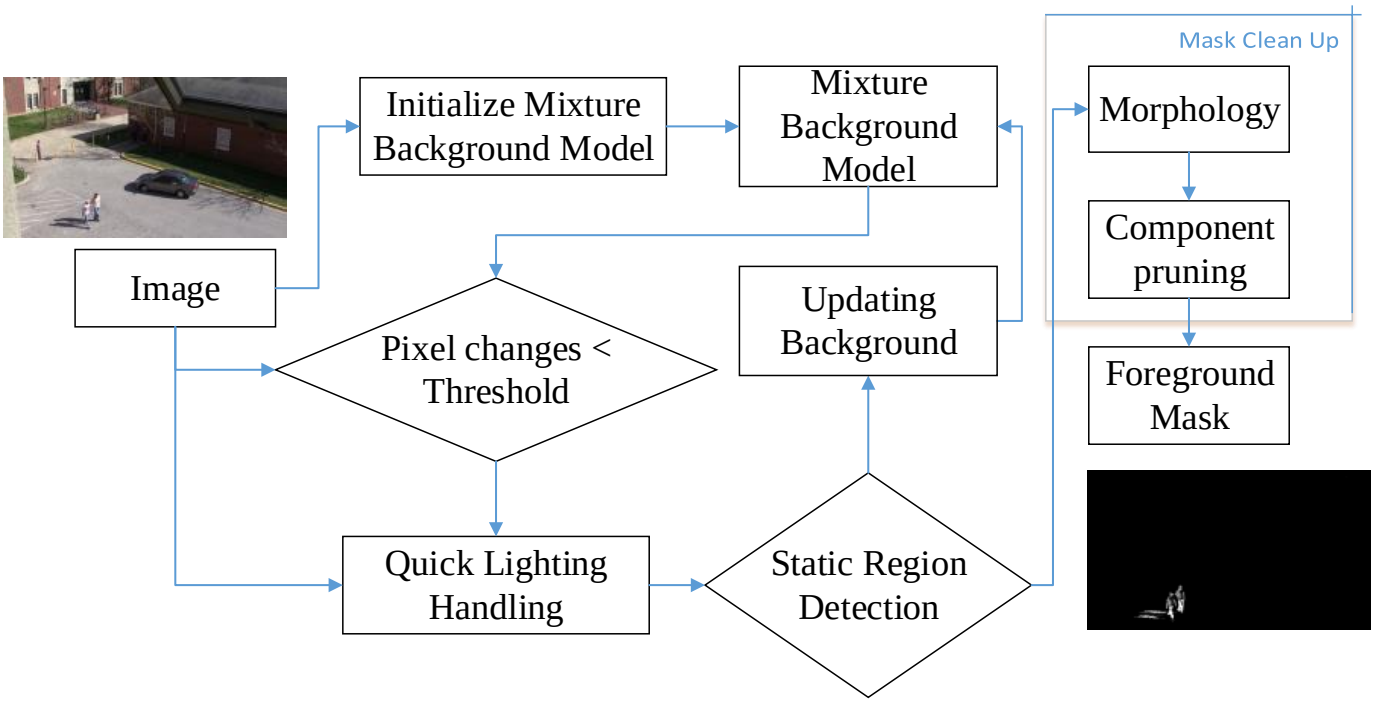
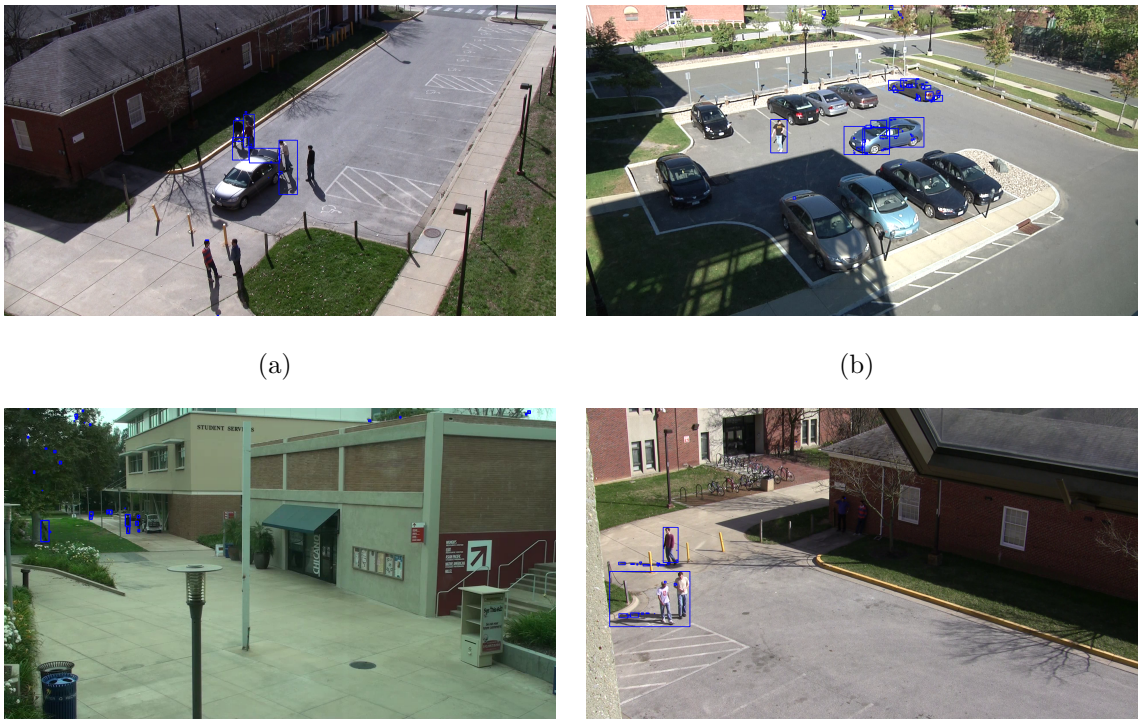


Figure 3.3: Adaptive Background Subtraction by Mixture Gaussian Models.

that it selects the appropriate number of Gaussian distributions for each pixel, providing better adaptability to varying scenes due to changes in the illumination. The process of subtracting the foreground mask is shown in Figure 3.3. The first frame of the video is used to initialize a Mixture Background Model. With this Mixture Background Model, which will be updated by the upcoming frames, the foreground mask can be extracted from the background image, with morphology and component pruning.

The whole process of foreground detection is as follows: Firstly, resize the image to a smaller resolution to reduce the processing time. The resizing factor is obtained through experiments reported in Section 5.2. Secondly, determine the foreground mask. The foreground is a binary image with pixel value 1 in foreground areas and pixel value 0 in background areas. Finally, find the contours in the binary image and resize the foreground mask to the original size of the image. These contours contain the foreground objects.

Figure 3.4 shows the results of the foreground detection method, with blue rectangles around the moving objects. The algorithm is able to detect moving cars in Figures b and c,



(a) (b)

(c) (d)

Figure 3.4: Example images with foreground detection results in blue rectangles.

and moving humans in all four figures; however, it fails to detect humans standing still in Figure 3.4(a) and standing in the shadow of the house in Figure 3.4(d). To overcome this limitation and identify the foreground regions occupied by humans, we employ a human body detector.

3.2.2 Human Body Detection

The foreground detector alone is not sufficient to determine the sensitivity of the scene, because of the following limitations: (1) the foreground detector does not differentiate between humans and other moving objects, and (2) the background model is not able to detect humans who are not moving or are standing in the shadows. To circumvent these problems, the system also detects human bodies in the video. The body detector uses a Histograms of Oriented Gradient (HOG) descriptor to represent the humans [20]. They found that normalized HOG descriptors perform well when describing feature sets

of humans relative to other feature sets. Figure 3.5 illustrates the progress of feature extraction and object detection. The detector window consists of a grid of overlapping blocks, in which Histogram of Oriented Gradient feature vectors are extracted. In more detail, the method divides the image into small cells and calculates a 1D histogram of gradient directions for each pixel [60]. Then, a linear Support Vector Machine (SVM) classifies the resulting vectors into person or non-person. The detection window is scanned across the image at all positions and scales. The results of the body detection are saved in a binary image I_t^{BD}, which has pixels of value 1 in the areas occupied by human bodies, and of 0 otherwise.

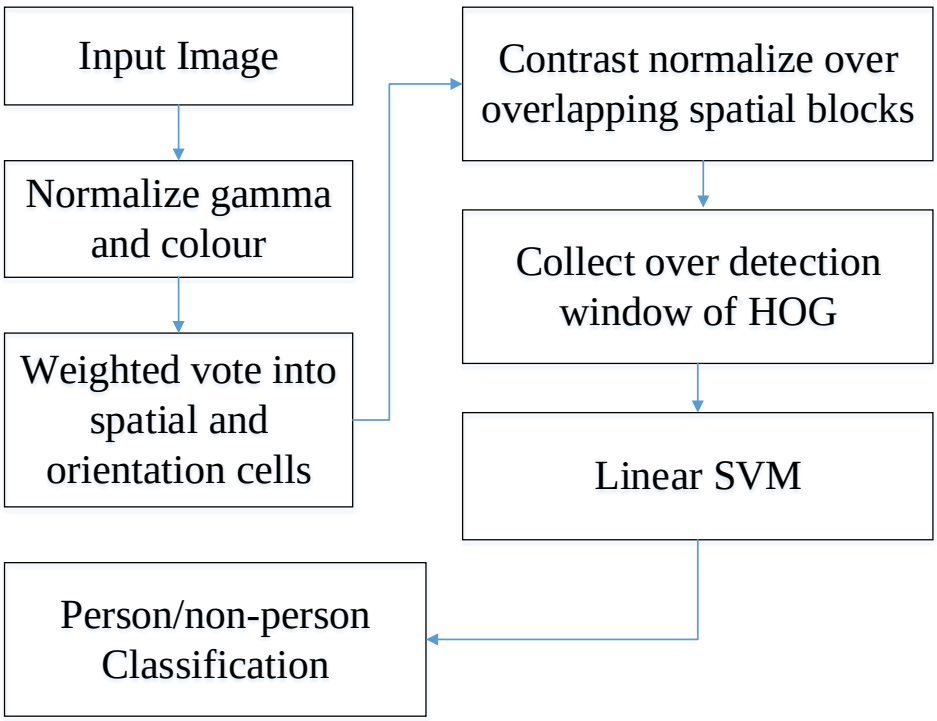


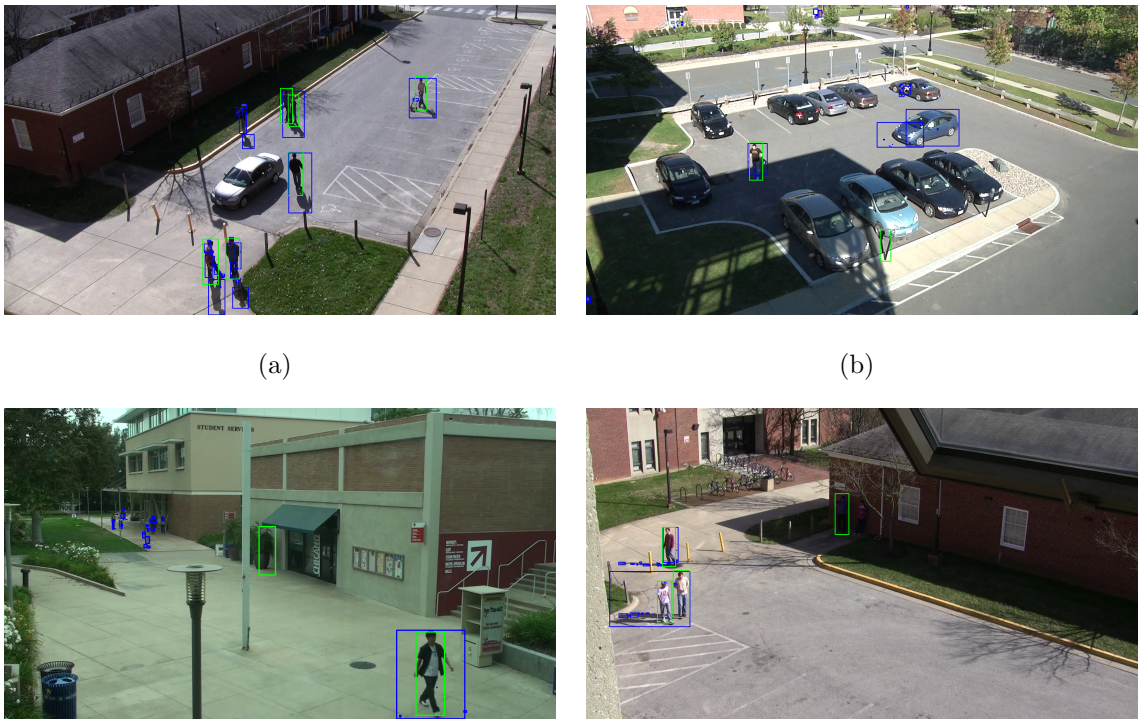
Figure 3.5: An overview of HOG

The whole process of human body detection is as follows:

- Resize the image to a smaller resolution to reduce the processing time. The resizing factor is obtained through experiments reported in Section 5.2.
- The human bodies are detected with the HOG descriptor class object. The initial coefficients for the linear SVM classifier are trained for people detection by OpenCV.

The human body mask is a binary image with a pixel value 1 where there are human bodies, and a pixel value 0 in regions where there are none.

- Cluster the overlapped functions and draw the corresponding results on the mask image. Finally, resize the mask image to the original image size.



(a) (b)

(c) (d)

Figure 3.6: Example images of human body detection method in green rectangles and blue boxes are the foreground objects.

The results of the body detection algorithm are shown in Figure 3.6 as green boxes. The examples overcome both limitations mentioned above. In Figure 3.6(a), the algorithm is able to detect the humans who are not currently moving, and in Figure 3.6(d), the algorithm successfully detects humans standing in the shadow of the house. Furthermore, the system is able to differentiate between humans and other moving objects, which is important because humans are more important for surveillance than other objects. Unfortunately false detection still cannot be avoided. In Figure 3.6(c), the potted plant is wrongly detected as a human body. In order to reduce the negative effects of false detection, we

combined foreground detection and human body detection.

3.2.3 Sensitivity Map

Combination

The result of each detection method is a binary image I, with the pixels belonging to a moving foreground object having a value of 1. The sensitivity map $\mathcal{S}$ for the frame is then calculated as

$$\mathcal{S} = I_t^{FG} + I_t^{BD}. \tag{3.1}$$

where I_t^{FG} and I_t^{BD} are the foreground and body of current frame , respectively. With foreground detection, a binary image I_t^{FG} is obtained with pixel value 1 in the regions belonging to the foreground, and 0 otherwise.

Calculating $\mathcal{S}$ this way automatically places greater emphasis on human bodies than on moving objects. For example, when a person is walking through the scene, the body is included in $\mathcal{S}$ twice. In other words, in the sensitivity map, the priority of the walking person increases, and the area where the body is detected will be lighter than before. Usually, the size of the region where a moving object like a car is detected is much bigger than a human body, which is likely to greatly affect the final display. However, the advantage here is that once the human body is detected, the region of the human body will be emphasized by both foreground detection and human body detection. It will therefore reduce the effect of the size of the detected region and increase the importance of the content in the surveillance video.

Accumulation

The system needs to detect the area of the video with the highest sensitivity over the course of ω frames. For this reason, the sensitivity maps for each frame in this time span are summed up into an accumulated sensitivity map $\mathcal{S}^{\omega}$. Accumulating the sensitivity also eliminates spuriously detected human bodies, and reduces the impact of noise in $\mathcal{S}$.

Once $\mathcal{S}^{\omega}$ has been accumulated over ω, which is the number of frames in the video, it is passed on to the ROI selection component. The experiment results of optimizing accumulation time is in Section 5.1. ω is a constant value that will remain the same after we calculate it.

3.3 ROI Selection

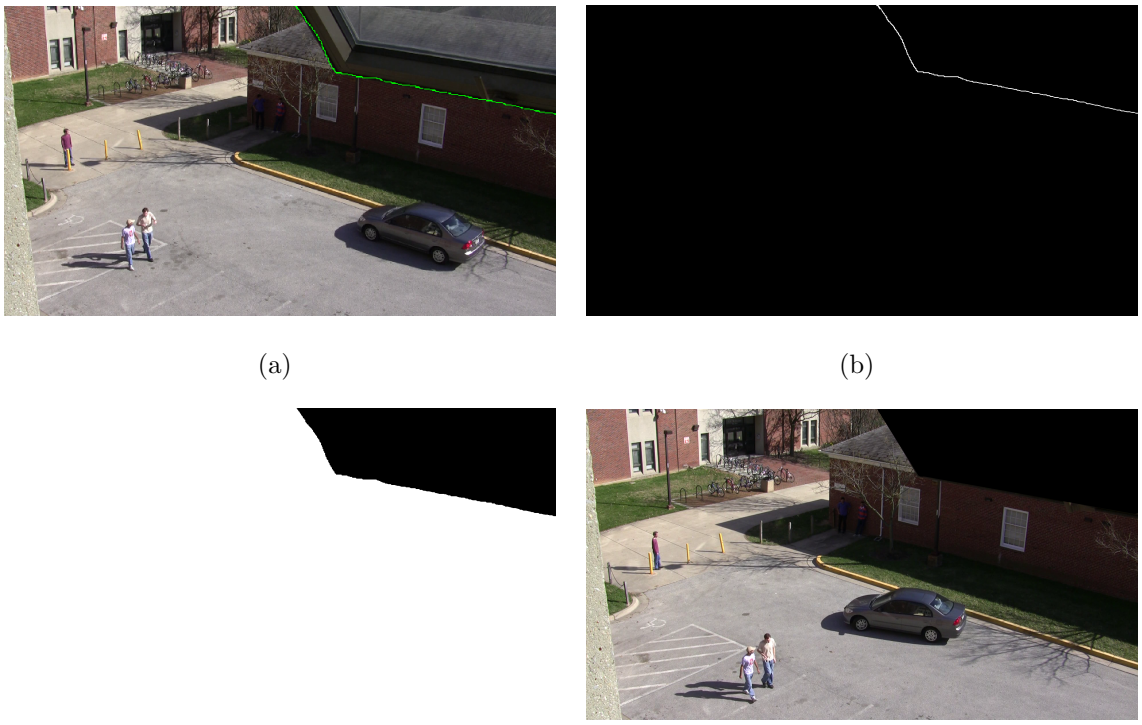
In this step we use the sensitivity map obtained in the previous step to determine the ROI for zooming. The sensitivity map identifies regions that contain moving objects, humans, and faces. Two problems occur when determining the ROI solely based on the sensitivity map: 1) Some regions may contain high motion, e.g., a road with traffic, and are therefore not relevant for surveillance. 2) In a relatively static scene, the same region may get selected repeatedly, reducing the systems coverage.

The system circumvents these problems by fusing the sensitivity map with the users input and the penalty map, and determines ROI by ranking candidate regions as described later.

3.3.1 User Input

The sensitivity map $\mathcal{S}^{\omega}$ as defined above cannot always identify the most important elements of the scene. Importance also depends on context, which is difficult to assess algorithmically. In a hallway, for example, it is more important to focus on the entrance than on the walls. A surveillance video of a parking lot might also include parts of the adjacent road, which could be outside of the scope of surveillance. It is thus important to include user input into the decision making process. User input is modeled as a map $\mathcal{U}$ of the same size as $\mathcal{S}^{\omega}$, which offsets the sensitivity. Since this information is mostly static, it only needs to be defined once, when the system is set up.

The boundary fill algorithm is used to fill the white connected component space, which starts from the seed point with the specified color. The connectivity is determined by the



(a) (b)

(c) (d)

Figure 3.7: Example images of setting user input.

color/brightness closeness of the neighboring pixels. The pixel at (x, y) is considered to belong to the repainted domain if:

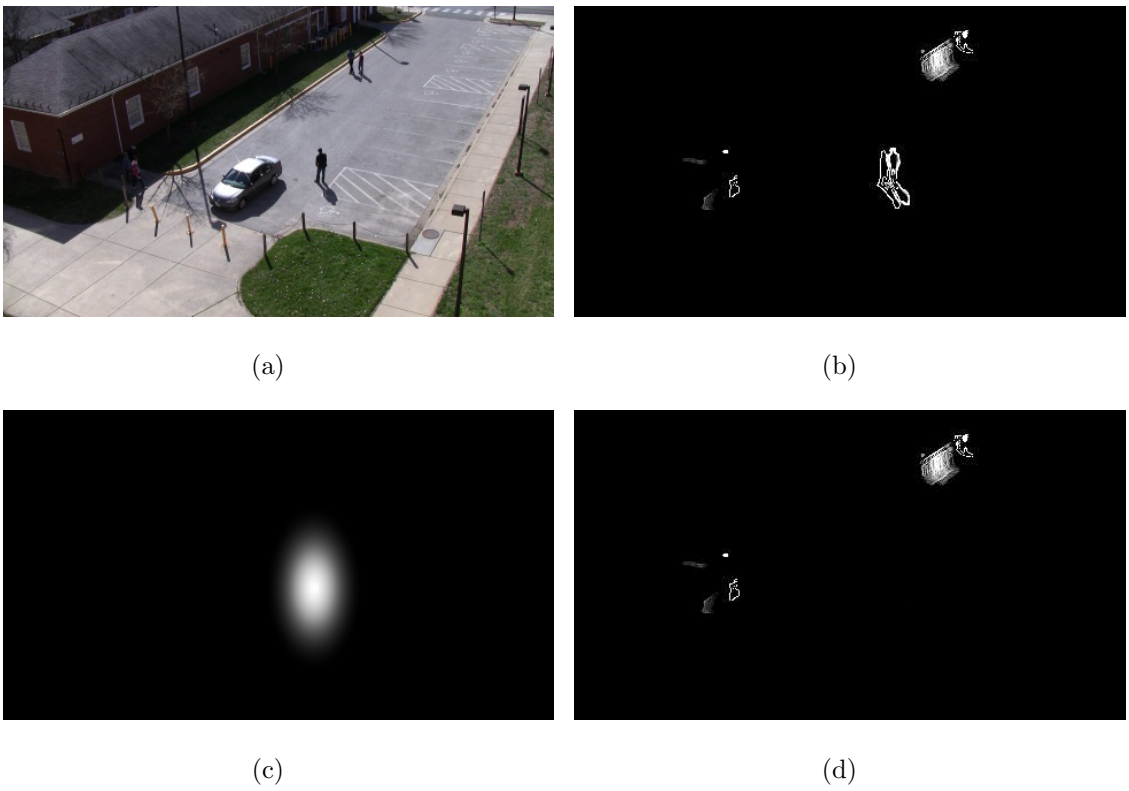
$$Mask(x', y') - loDiff \leq Mask(x, y) \leq Mask(x', y') + upDiff \tag{3.2}$$

where $Mask(x', y')$ is the value of one of the pixel neighbors that is already known to belong to the component, and $loDiff$ and $upDiff$ are difference of up and down, respectively.

The flow of the setting user input process is shown in Figure 3.7. Once the user starts manually removing unwanted regions, the system will stop at the first frame received from the surveillance camera. The user can then simply manually draw the outline of those regions. For example, the window in Figure 3.7(a)) has been outlined in green. The corresponding mask image will be output at the same time, which is like Figure 3.7(b). Then, users can just click on the region they wish to keep. Once selected, the specified region will be filled with white, meaning that this part is useful, whereas the black one is not. Figure 3.7(c) is the input mask image that will always work on the final decision map

in Section 3.3.3.

3.3.2 Penalty Map



(a) (b)

(c) (d)

Figure 3.8: The processing flow of penalty

The purpose of this system is to automatically provide the operator with the most important context when watching the video. However, if a region is identified as being more important than everything else in the video in a specific frame, it will probably also be the most important area in the later frames; this will result in the same region being detected and displayed more than once. To control the decay of the specified region over time, we give a penalty to the sensitivity of a region before it is presented to the operator.

The penalty map $\mathcal{P}_t$ has float values between 0 and 1, with higher values meaning a higher penalty. A Two-dimensional Gaussian function is calculated for the chosen ROI. The penalty map $\mathcal{P}_t$ is defined as

$$\mathcal{P}_t(x,y) = \begin{cases} Aexp\left(-\left(\frac{(x-x_0)^2}{2\sigma_x^2} + \frac{(y-y_0)^2}{2\sigma_y^2}\right)\right) & if\ t\ is\ the\ 1^{st}\ frame\ of\ \omega \\ \mathcal{P}_t(x,y)\ *\ \lambda & \lambda \in [0,1],\ otherwise \end{cases} \tag{3.3}$$

where the coefficient $A = 1$, which is the pixel range of the binary image, X_0,Y_0 are the center coordinates of the selected ROI, σ_x,σ_y are the width and height, respectively, of the selected ROI.

The penalty mechanism will act on the selected region of the sensitivity map $\mathcal{S}^\omega$ before the system decides to zoom-in to another region. This Gaussian function is added to $\mathcal{P}^\omega$ in every frame, when a new $\mathcal{S}^\omega$ is available. The process of penalizing a previously chosen ROI is illustrated in Figure 3.8. The top left image shows a frame from a surveillance video. The top right frame shows the accumulated sensitivity map $\mathcal{S}^\omega$. The person in the center gets chosen as the next ROI. As a result of the chosen ROI, the penalty map on the bottom left is updated. After ω frames, the previously chosen region is penalized in the sensitivity map and will not be chosen again (bottom right).

As an example, if there are two sensitivity map, one is $\mathcal{S}_1^\omega$, and the other is $\mathcal{S}_2^\omega$, where $\mathcal{S}_1^\omega$ is formed before $\mathcal{S}_2^\omega$. After the selected region is obtained from $\mathcal{S}_1^\omega$, the corresponding position of this region is used as a penalty region in the new sensitivity $\mathcal{S}_2^\omega$. Every ω frames, when a new ROI is selected, the penalty map is multiplied by a factor λ to decrease the penalty over time, which is defined in the second row of formula 3.3.

The sensitivity of a selected ROI is thus reduced at first, and then slowly increases back to its original value.The purpose of using a two-dimensional Gaussian is to ensure that the center of the object remains significant, even if there is a penalty on the region so that it is not displayed continuously. Furthermore, after penalizing the specified region for a period of time, it doesn't mean that this region is not important any more; in fact, the sensitivity map formed by the accumulation of ω frames can still keep the detected regions importance information. In a typical case, there is only one moving object that appears in the observed scene. The only detected region that will be chosen is around that person, so the presentation part should keep showing this area continuously, until it changes.

3.3.3 Fusion and Ranking

The decision map $\mathcal{D}_t$ is defined as

$$\mathcal{D}_t = (1 - \mathcal{P}_t) \cdot (\mathcal{S}_t \cdot \ \mathcal{U}). \tag{3.4}$$

where $t \in \omega$, all operators used here process the maps is pixel-wise, $\mathcal{P}_t$ is penalty map, $\mathcal{S}_t$ is the sensitivity map, and $\mathcal{U}$ denotes user input.

For the purpose of ROI detection, $\mathcal{D}_t$ is converted into a binary image by applying a low threshold θ, which is defined as

$$\mathcal{D}_t^{dst}(x,y) = \begin{cases} 0 & if \ \ \mathcal{D}_t^{src}(x,y) < max(\mathcal{D}_t^{src}(x,y)) * \theta, \\ 1. & \end{cases} \tag{3.5}$$

where $\mathcal{D}_t^{src}(x,y)$ is the original sensitivity map, θ is a constant float value, and $max(\mathcal{D}_t^{src}(x,y))$ represents the maximum pixel value in the sensitivity map.

The morphological operations Erode and Dilate are then used to reduce noise in the map. Since the detection results from background subtraction and human body detection are both rectangular, in some case, there are several distinct rectangles representing the same detected object. It is therefore necessary to assemble these rectangles. The method used to find the convex hull is introduced here, which acts on the $\mathcal{S}_t$.

Next, the system extracts the extreme outer contours from the white areas in the binary image and merges the adjacent contours, if necessary. The algorithm used in this system to find contours is proposed by [69]. The bounding boxes of the connected contours are the potential regions of interest in which we need to zoom. For each potential ROI, the decision value will be calculated by the sum of all the values inside the ROI in the original unthresholded $\mathcal{D}_t$. This allows for the ranking of the potential ROIs according to the amount of sensitivity they contain. The ROI with the highest decision value is selected and sent to the presentation component.

3.4 Presentation

Once the system receives the final selected region from the sensitivity map, which is accumulated over ω frames through several procedures, it will begin the presentation part. The selected region will be used by RGB tracking to predict the behavior of the target in the next frame. The method of tracking we use is mean shift, which is a very popular tracking method, whose efficiency and robustness greatly benefit the system. In this system, the tracking method presents the video context around the object, but also plays a very important role in the reduction of the processing time of the whole system. The overall presentation time is $\triangle$ frames, which includes accumulation time ω. The tracking

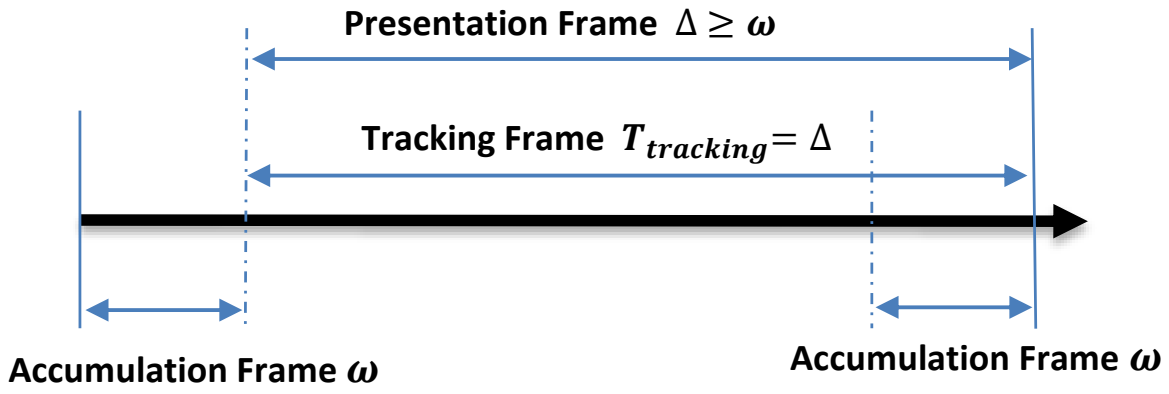


Figure 3.9: Presentation time.

method functions on the next $\triangle$ frames, while the visualization part displays the frame that will be sent to the remote controller. In the visualization part, the display flow starts by zooming into the specified region. Then, it keeps tracking the object in this region, later zooms out of the region, and finally, keeps a global view of the scene. The duration of each step is different, and is based on our experience with zooming. The presentation details are shown in Figure 3.9, which shows the complete flow of how the presentation works. When the system starts, ω frames are combined to form a sensitivity map, which will cause a little sustainable latency.

3.4.1 ROI Tracking

The object is usually moving, which will leave the zoomed in window during the course of the Delta frames. In order to solve it, a tracking mechanism is added in our system.

One of the features of ROI tracking is the ability to keep the object in a detected region, which will always be in focus. In other words, simply zooming into a fixed region will lead to false detections and missing video context parts, for example, when a detected object moves out of the selected region. When the tracking mechanism is introduced, the screen always follows and displays the selected object, thus ensuring that most of the information around the object is captured. After using the tracking method and comparing it with others systems, the total running time for image processing containing tracking is reduced significantly, which is proven in Section 5.4.

The output of the ROI selection step is a rectangle, which is likely to contain the objects of interest. We employ the mean shift tracking algorithm to track objects in each RGB frame sequence ($\triangle$) during the zoom period [14]. Mean shift is a powerful non-parametric iterative algorithm that can be used for various purposes, for example, clustering. Fukunaga and Hostetler [29] were the first to propose this method, which is now present in other fields like computer vision.

Mean shift considers the feature space as a probability density function (PDF). Dense regions correspond to local maxima or modes. For each data point, the gradient ascent is performed on the local estimated density until convergence. Assuming $g(x) = -K^{'}(x)$, the mean shift is defined as

$$m_h(x) = \frac{\sum_{i=1}^{n} x_i g(||\frac{x-x_i}{h}||^2)}{\sum_{i=1}^{n} g(||\frac{x-x_i}{h}||^2)} - x \tag{3.6}$$

where the kernel $k(X)$ satisfies $K(x) = c_{k,d}(||x||^2)$.

This procedure can be summarized as the following: For each point x_i, compute mean shift vector $m(x_i^t)$ and move the density estimation window by $m(x_i^t)$ repeatedly, until convergence. In more details, mean shift associates each data point with the nearby peak

of the dataset's PDF, defines a window around it and computes the mean of the data point. It then shifts the centroid of the window to the mean and repeats the algorithm until it converges. After each iteration, we can see that the window shifts to a denser region of the dataset. In OpenCV 2.4.9, in order to increase the speed of the calculations, the kernel is defined as Rectangular.

$$\phi = \begin{cases} 1 & a < x < b \\ 0 & else \end{cases} \tag{3.7}$$

At the implementation level, the tracking method functions on each frame and the predicted ROI will be the new target region to track in the next frame. This method is an iterative scheme based on the comparison of the histogram of the original object in the current image frame and the histogram of candidate regions in the next image frame. The aim is to maximize the correlation between the two histograms. Figure 3.10 shows a flow chart of how mean shift works when tracking a colorful object. Figure 3.11 shows the intermediate images during processing. The moving object is obtained by the ROI selection part. The chosen area is defined as ROI, which is the initial search window. Back projection is used as a pixel of color probability distributions, which calculates the color histogram of the detected sensitivity part in the selected ROI. The HSV space denotes color information with H weight. During the calculation process, all the RGB pixels should be transformed to an HSV space and the H value is used to compute a 1D histogram of an image. The whole process is defined as back projection. The information about the selected ROI's position and size is used to calculate the centroid of the window and set the center of the window at the centroid. This step is repeated a specified number of times, until the center of the window is in convergence.

The adjusted region is a bigger region than the selected ROI, whose size will be used as the input image size when calculating back projection. The reason is that considering the normal situation of moving people and cars, it is reasonable to only calculate the back projection around the selected ROI. The advantage is that the processing speed is much faster than when calculating the back projection of the whole image, because the tracking

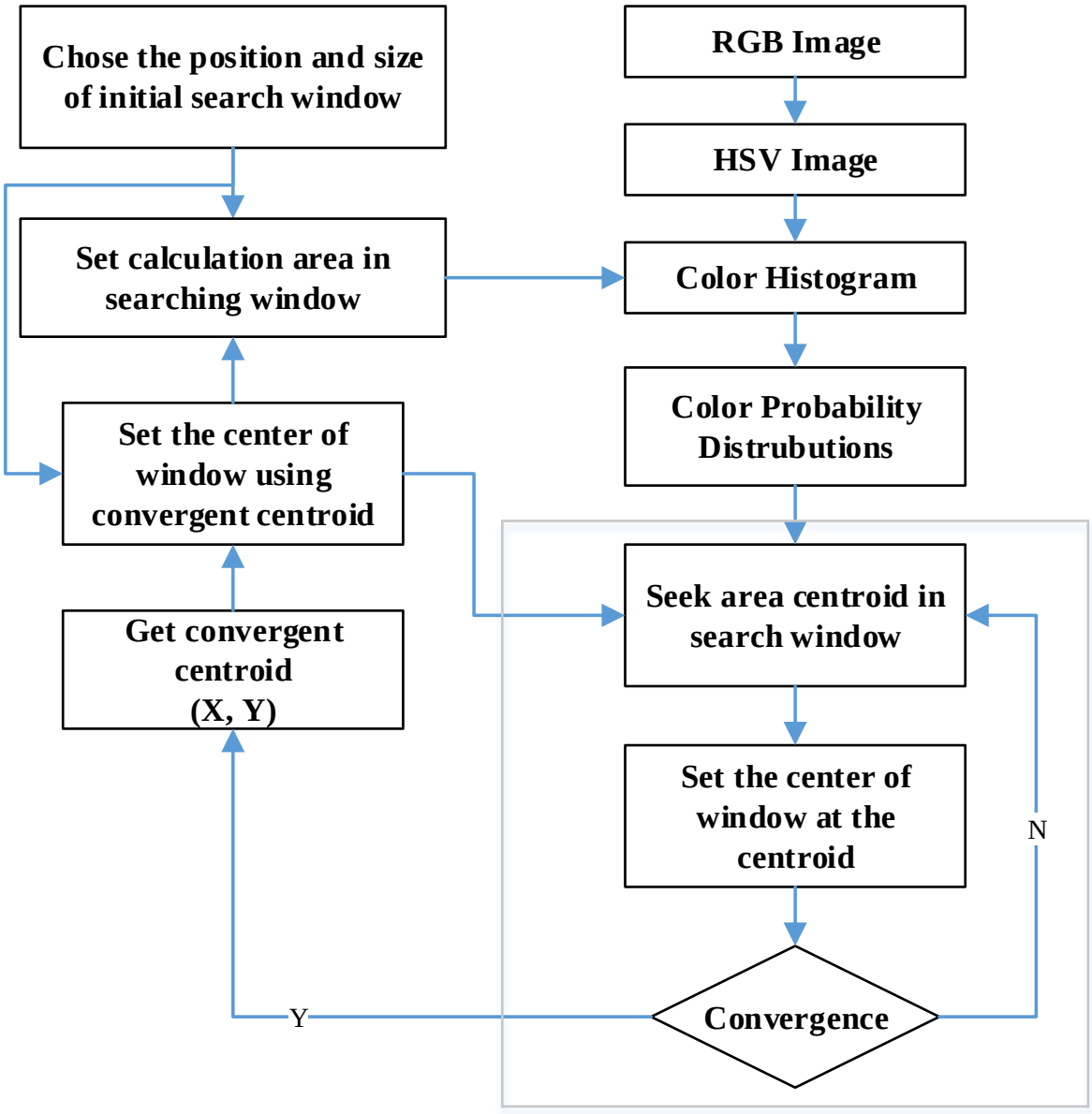


Figure 3.10: Flow chart of how mean shift algorithm works on tracking

method is faster than two detection methods for each frame.

In the system, after we get the new predicted region, the returned ROI is not stable for visualization. Shaking affects the experience of video monitoring. There is an InRange mask in Figure 3.11, which is a simple threshold method. This method is mainly used to reduce the excessive light and illumination in our system, in order to prevent certain

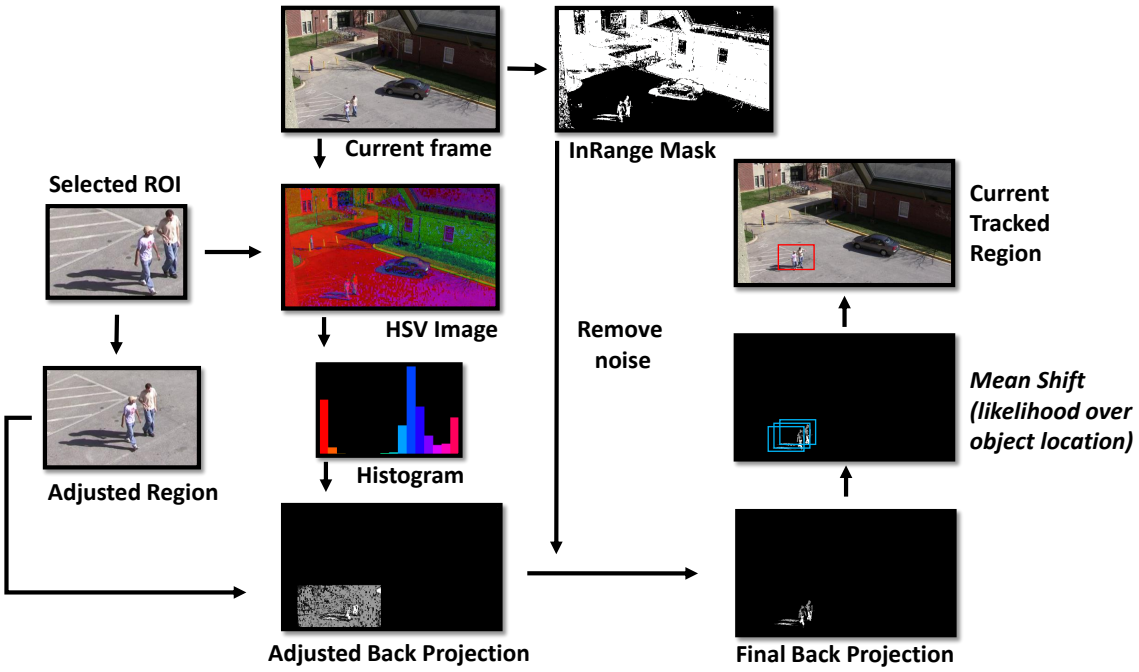


Figure 3.11: Flow chart of tracking displayed by image

false values. This threshold method is not appropriate for all scenarios, which is why the stabilization part below is introduced to avoid this serious problem.

3.4.2 Stabilization

Due to shaking, and in order to improve the operator's monitoring experience, a median filter is introduced to reduce the effects of an unstable presentation.

The main idea behind the median filter is for it to run through the input signals. Each new adjusted signal comes from the median of neighboring signals by sliding signal by signal. The pattern of neighbors is usually named "window". There are two possibilities for the number in the window; one is odd, the other is even. For an odd number of elements in the window, the middle value will be selected after sorting the window numerically. For an even number of elements in the window, there is more than one possible median strategy. In our system, we consider the rule as $MEDIAN = (n+1)/2$, where n is the number of elements in the window. The center of each predicted ROI received from the tracking part

are saved as elements in the window, which has a fixed size ω. The corresponding width and height of each region is also saved.

The flow of stabilization is a dynamic process and the values of the x-axis and the y-axis are processed separately by the median filter.

For every input,

1. If the window size ω is not achieved, save the value of the input.
   Otherwise, go to step (2).
2. Save the value of the input that replaces the first value received.
3. Sort the window numerically.
4. Choose the median value that will be the output.

Then, let the center of the selected region and its corresponding width and height be the final detected region for the presentation part. However, since the image displayed on the screen should have a fixed scale, the ROI aspect ratio adjustment is applied to solve this problem.

3.4.3 ROI Aspect Ratio Adjustment

At the beginning of each block of ω frames, the aspect ratio is given an ROI for the entire block. The frames of the block are contained in the buffer. Note that the ROI selection may give rectangular areas with an arbitrary aspect ratio. The aspect ratio $\phi = w/h$ of the ROI is generally not identical to the target aspect ratio ϕ_t. $\phi < \phi_t$ means that the chosen ROI is too high. We thus increase the width of the ROI as $w = \phi_t h$, so that the aspect ratios match again. The case $\phi > \phi_t$ is handled analogously. This guarantees that the selected region is fully contained in the created view.

3.4.4 Visualization

The presentation of a block of frames always starts with a fully zoomed out overview. Within the block, it zooms into the ROI, stays zoomed in for a certain amount of time, and then zooms out again. At the end, it stays on the overview image for a short period of time.

Zooming is implemented as an interpolation between the parameters of the full frame and the chosen ROI, but simply zooming into the ROI from the full overview will affect the viewing experience of the operator. Consequently, the cubic Hermite spline is introduced for smooth zooming, instead of using a linear equation. The cubic Hermite spline is a spline where each piece is a third-degree polynomial specified in Hermite form[83], which is very easy to calculate but also very powerful. It is efficient and only occupies a very small portion of the total processing time. It is often used to smoothly interpolate between key-points (like object movement in keyframe animation or camera control). Figure 3.12 shows the complete flow of zooming. The cubic Hermite spline works in the two stages: zoom in and zoom out, individually. For each rectangle, there are four features: the value of the x-axis, the value of the y-axis, the value of the width, and the value of the height, which are defined as:

$$\begin{bmatrix} x' \\ y' \\ h' \\ w' \end{bmatrix} = (2t^3-3t^2+1)\begin{bmatrix} x_{start} \\ y_{start} \\ h_{start} \\ w_{start} \end{bmatrix} + (t^3-2t^3+t)m_0 + (-2t^3+3t^2)\begin{bmatrix} x_{end} \\ y_{end} \\ h_{end} \\ w_{end} \end{bmatrix} + (t^3-t^2)m_1. \quad (3.8)$$

where $t = \frac{the\ \ number\ \ of\ \ current\ \ frame}{the\ \ number\ \ of\ \ presentation\ \ time}$, m_0 is a starting tangent and m_1 is an ending tangent, which are both 0, x is the x-axis of top-left corner of rectangle, y is the y-axis of the top-left corner of the rectangle, h is the height of the rectangle, and w is the width of the rectangle.

Figure 3.13 shows a set of images for a whole period of presentation from video D. The zoomed target in this period is a blue car. The course of the zoom-in can be seen from Figure 3.13(a) to Figure 3.13(f). The period of stay can be seen from Figure 3.13(g) to

Figure 3.13(l). The course of the zoom-out and stay can be seen from Figure 3.13(n) to Figure 3.13(t).

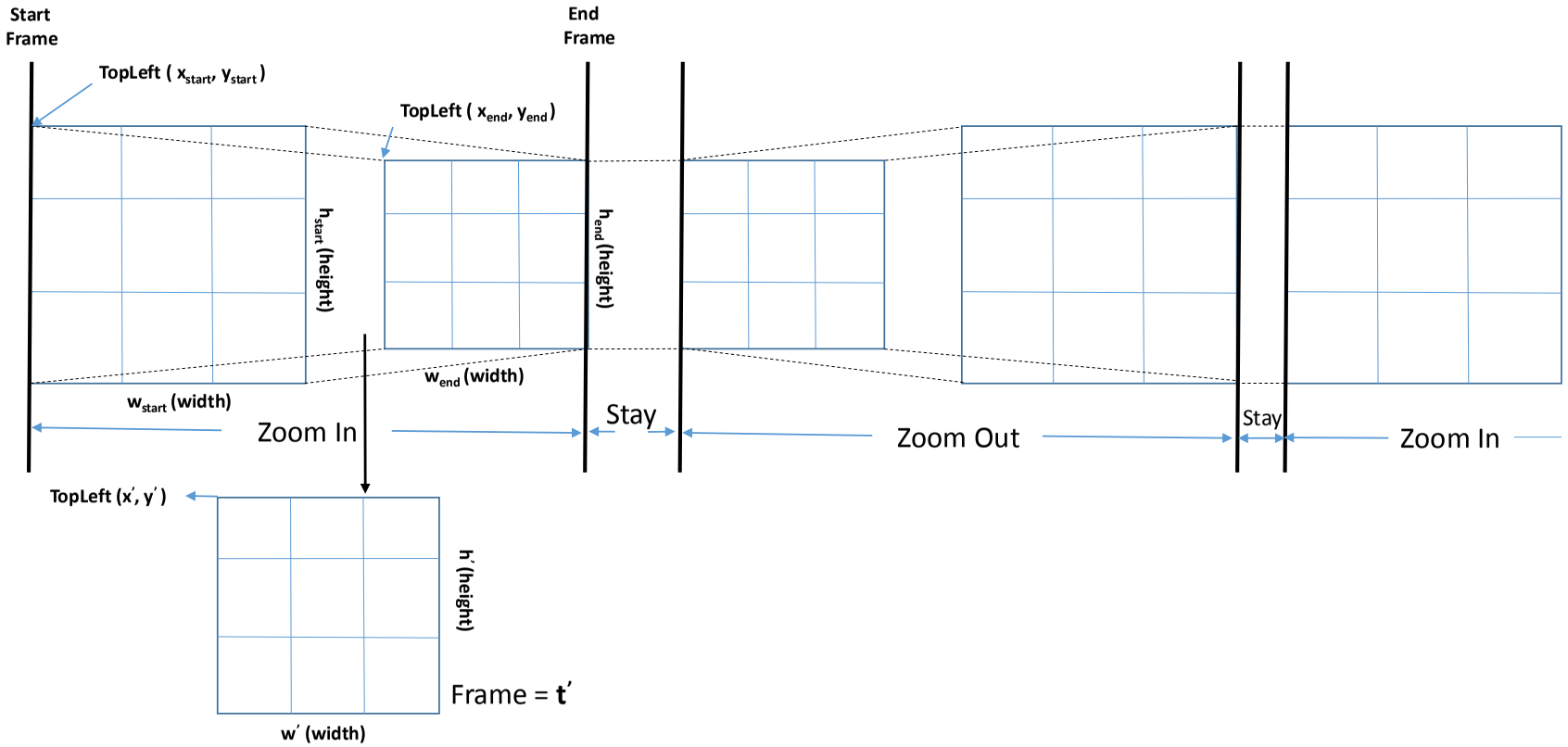


Figure 3.12: Flow of zooming.

(a) (b) (c) (d)

(e) (f) (g) (h)

(i) (j) (k) (l)

(m) (n) (o) (p)

(q) (r) (s) (t)

Figure 3.13: Example images of zoom-in, stay, zoom-out and stay

Chapter 4

System Implementation

In this chapter, the platform and deployment of the proposed system is introduced, especially the software layer structure. Then, the whole systems architecture is shown with a complete system flow chart.

4.1 Platform and Deployment

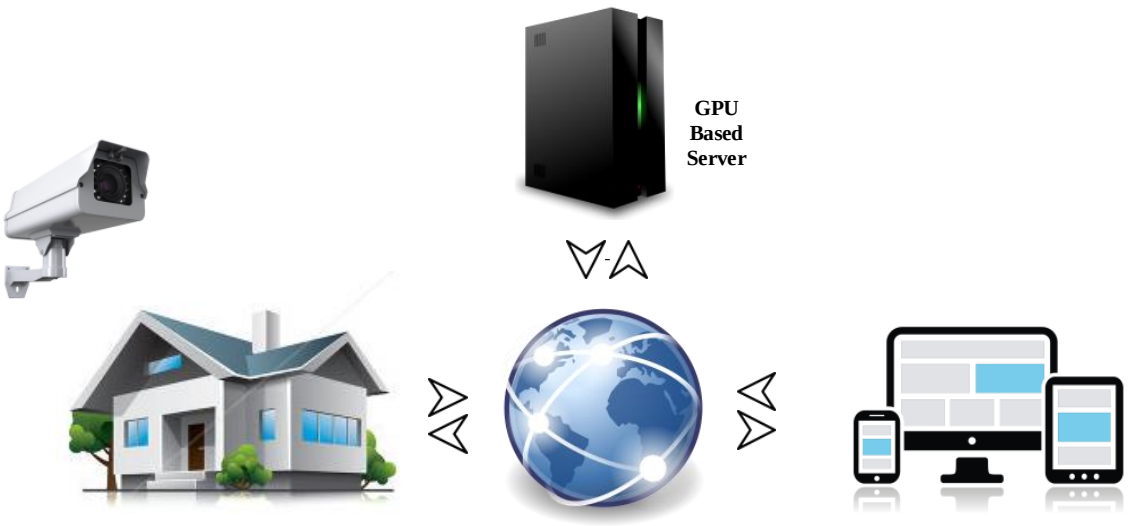


Figure 4.1: System Deployment

CPU	3rd Generation Intel® Core™ i7-3520M (3.60 GHz, 4MB L3, 1600MHz FSB)
Memory	6GB 1600MHZ
Graphics	Intel® HD graphics 4000
Storage	500GB (7200rpm)
WIMAX/WIFI	Intel® Centrino® Advanced-N 6205 AGN

Table 4.1: PC configuration table

Software Layer

The whole SSA system is programmed in C99 & C++11, based on several third-party libraries under Ubuntu 14.04 LTS operating system, which is shown in Figure 4.2. The detailed information of these libraries and their usage is provided as follows:

OpenCV 2.4.9

Open Source Computer Vision Library (OpenCV) is an open source computer vision and machine learning software library [1]. It mainly focuses on realizing real-time computer vision. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in commercial products. There have been contained more than 2500 optimized algorithms, including classic and state-of-the-art computer vision and machine learning algorithms. The current version of OpenCV is 3.0 BETA, but 2.4.9 is a mature version using complete oriented-object programming. In the system, most of the algorithms such as foreground detection, human body detection, image resize etc. refer to it.

FFmpeg 2.6.1

FFmpeg is the leading multimedia framework that produces libraries and programs to handle multimedia data such as decode, encode, transcode, mux, demux, stream, filter [2]. It supports the most obscure ancient formats up to the cutting edge formats, no matter where they were designed. In the system, FFmpeg has been assembled in OpenCV to handle

[1] http://www.opencv.org

[2] http://www.ffmpeg.org/about.html

Figure 4.2: Software layer of system

video coding. The methods of generating or importing video (images) are all supported by FFmpeg's. The FFmpeg development team recommends the use of the latest version, which is version 2.6.1.

Boost Library 1.5.7

Boost is a set of libraries for the C++ programming language, which provides support for tasks and structures such as linear algebra, pseudorandom number generation, multi-threading, image processing, regular expressions, and unit testing [3]. It contains over eighty individual libraries, which are aimed at a wide range of C++ users and application domains. The Boost.Filesystem library is convenient to be learned and used regardless of

[3]http://www.boost.org

the operating system [4]. In our system, we mainly use Boost libraries for file I/O evaluation.

CUDA 7.0

CUDA is a parallel computing platform and programming model made by NVIDIA. It enables dramatic increases in computing performance by harnessing the power of the graphics processing unit (GPU). CUDA gives developers direct access to the virtual instruction set and memory of the parallel computational elements in CUDA GPUs [5]. OpenCV is pre-built on the CUDA whose set of classes and functions utilizes GPU computational capabilities [6].

OpenCL 2.1

Open Computing Language (OpenCL) is a cross-platform for parallel programming, consisting of the central processing units (CPUs), graphics processing units (GPUs), digital signal processors (DSPs), field-programmable gate arrays (FPGAs), and other processors. Developers are able to start compute kernels written through a limited subset of C programming language on a GPU. OpenCL greatly accelerates the speed of gaming and entertainment applications in various market categories [7] [8]. OpenCV is pre-built on it.

OpenMP 4.0

Open Multi-Processing (OpenMP) is an implementation of multiprocessing, a method of parallelizing whereby a master thread forks a specified number of slave threads. The system then divides a task list among them, after which the threads run concurrently, with the runtime environment allocating threads to different processors. It consists of a set of compiler directives, library routines, and environment variables that influence run-time behavior [9]. OpenCV is pre-built on it.

IPP 8.0

[4] http://www.boost.org/doc/libs/1_57_0/libs/filesystem/doc/index.htm
[5] http://www.nvidia.ca/object/cuda_home_new.html
[6] http://docs.opencv.org/modules/gpu/doc/introduction.html
[7] http://www.khronos.org/opencl/
[8] http://developer.nvidia.com/opencl
[9] http://openmp.org/wp/

Intel Integrated Performance Primitives (Intel IPP) is an extensive multi-threaded software library of software functions to help you develop multimedia, process data [10]. OpenCV is pre-built on it.

4.2 Development of System Architecture

In this section, the system is represented by two UML diagrams, a sequence chart and a flowchart.

The sequence chart shows a complete period of how each image coming from video converts to the display image for visualization, which is shown in 4.3. Once the system gets the video input from the surveillance camera, the video is dealt with frame by frame. Foreground detection and human body detection are used to detect the informative region on the input. The output of these two methods is a binary image. The binary image is converted to a gray image, which will be accumulated over Accumulation Time, which means that the two detection methods function on each frame only during the accumulation time. The accumulation result names sensitivity map. Subsequently, the detected region is calculated based on the sensitivity map, by finding the contour in the image and the maximal sum of pixels in the region is returned as the selected region.

The selected region will be used to calculate the histogram and as one of the input parameters of the Mean Shift algorithm. Then, according to the histogram of the selected region and the CSV format of the input image, the back projection can be calculated. The meanshift algorithm will predict the position of the target object in a new frame. The stabilization is then adopted to stabilize the shaky image problem. The Cubic Hermite Spline is split into three stages: zoom-in, stay, and zoom-out, based on the number of frames in the video. Finally, the display images will be returned to the remote mobile devices. Greater details of the flow are shown in Figure 4.4.

[10] https://software.intel.com/en-us/intel-ipp

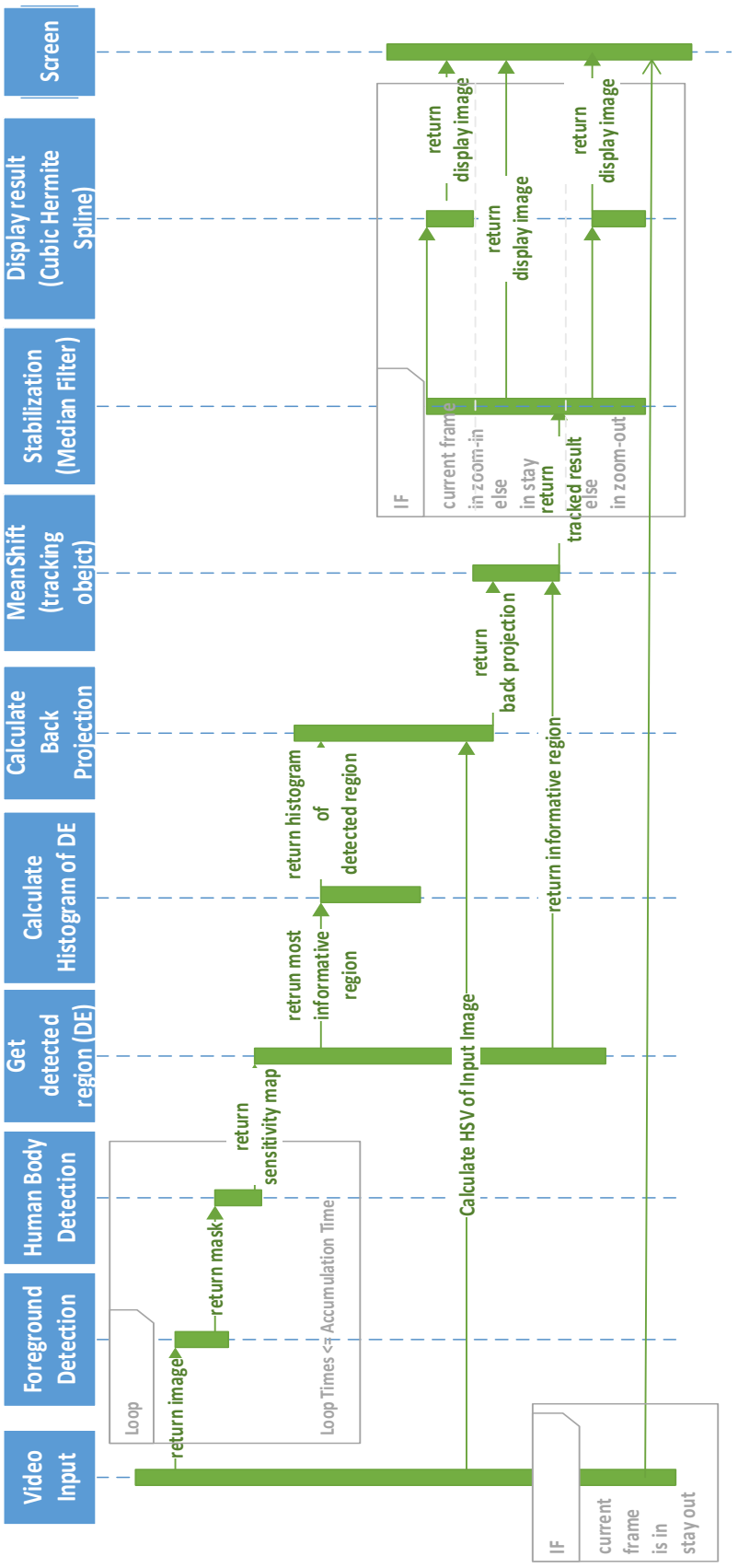


Figure 4.3: Sequence chart of whole system.

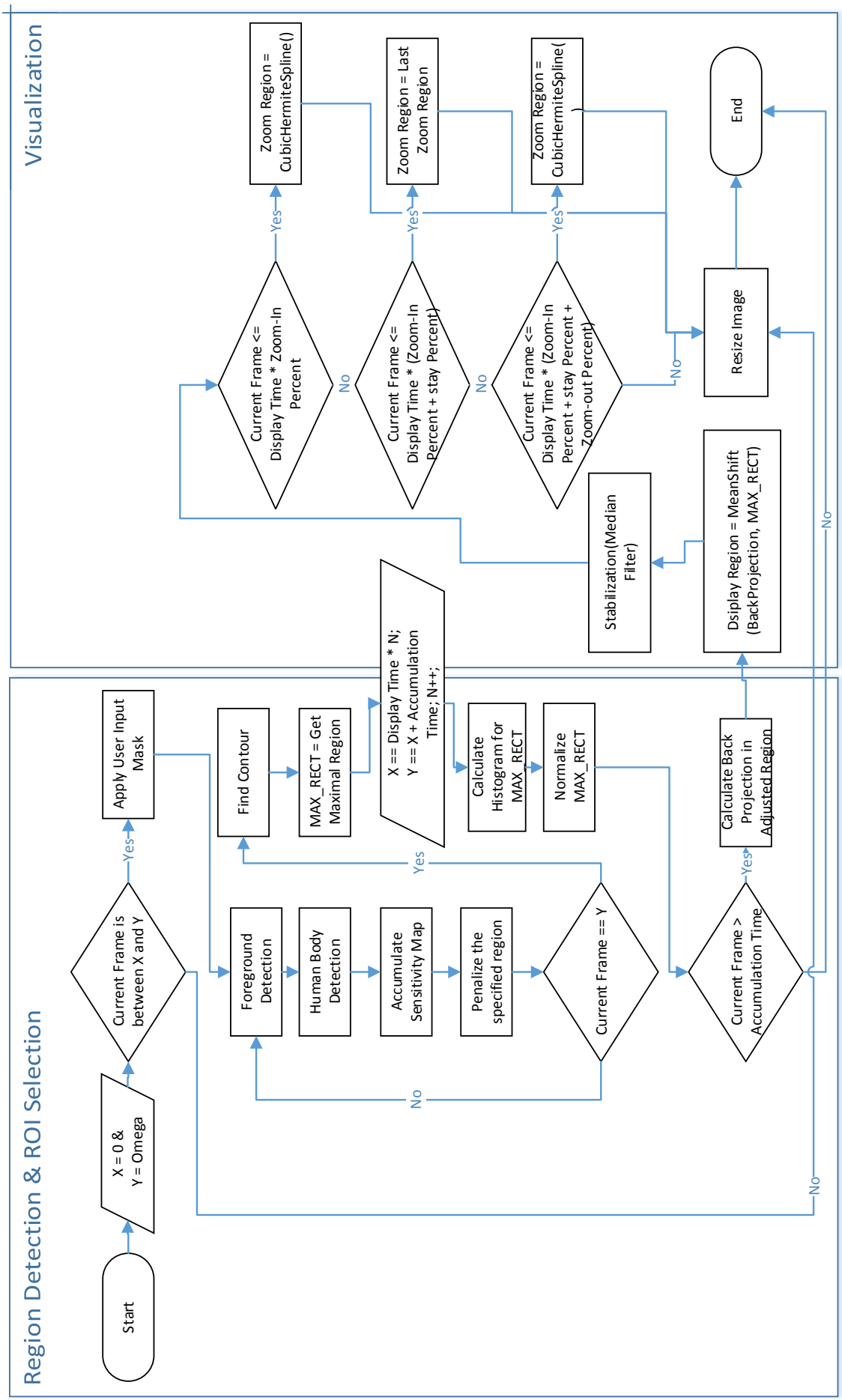


Figure 4.4: Processing flowchart of whole system.

Chapter 5

Evaluation and Results

There are three parts in this chapter. In the first part, we conducted experiments to obtain the optimal value of the accumulation window ω. The goal of this part is to find an optimized accumulation time for both foreground and human body detection in one video scene, while maintaining accuracy. In the second part, we analyzed the timing performance of different components and found the minimum image resolution required for each processing step (body detection, and foreground detection). The goal of this experiment is to reduce the processing time without compromising quality. Since the proposed system contains a number of features that are hard to be compared directly, in the third part, we assessed the effectiveness of the proposed system for the surveillance task through a user study. We divided the user study into two small parts. First of all, a simplified system extracted from the proposed system was compared with two other approaches. Secondly, the proposed system was compared with a scaled system only to see how its effectiveness.

The compared features of each system is listed in Table 5.1. The scaled system is the most common way to display surveillance videos on remote devices, by scaling the original image size to a lower specified one. The pan system only contains two detection methods and the display method adapts the pan way, where the former selected region shifts to the next selected region like a lens shifts from one point to another. The simplified system is almost same as the system we proposed, but the differences are that: (1) the simplified

System		Scaled	Pan	Simplified	Proposed
Region detection	Foreground detection	No	Yes	Yes	Yes
	Human body detection	No	Yes	Yes	Yes
	Optimized resolution	No	No	No	Yes
	Optimized accumulation time	No	No	No	Yes
ROI selection	User Input	No	No	Yes	Yes
	Penalty	No	No	Yes	Yes
Visualization	Tracking	No	No	No	Yes
	Presentation	Scaled	Pan	Zoom-in/Stay/Zoom-out	Zoom-in/Track/Zoom-out

Table 5.1: Experimental systems comparison

system doesn't have a method of optimizing accumulation time (2) the simplified system doesn't have a method of optimizing the resolution for each detection method (3) the simplified system doesn't contain a RGB tracking method.

Video Dataset

Four surveillance videos with a resolution of 1920×1080 and a length of one minute were used. They were selected from the VIRAT database, which was designed for performance assessments of activity detection algorithms [54]. Compared to other datasets, the selected videos were recorded in natural scenes which shows people doing regular actions, with complex backgrounds. The dataset provides the original videos with HD quality which are suitable for evaluating our system. Multiple sites in a variety of camera viewpoints and resolution are collected in the dataset. The representative frames of our selected four videos are shown in Figure 5.1. We can see that the selected four videos cover different scenarios in surveillance videos including regular activities, such as moving car at parking lot. People and cars are the main components in these scenes and the actions and events (e.g., number of moving persons or cars) are all normal cases in the videos.

5.1 Optimize Accumulation Time

We found that the regions of interest can be correctly detected for each frame, while false detections appear randomly among the frames. With this observation, we accumulated the

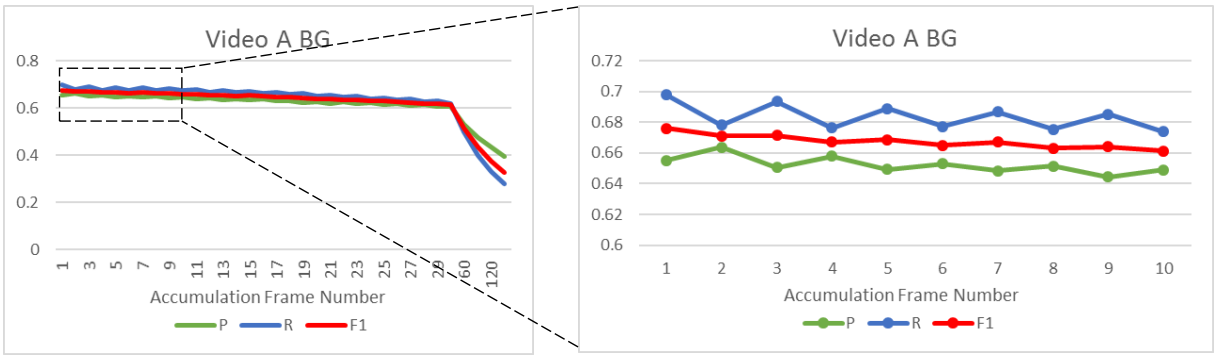
Figure 5.1: Example frames from the four scenes used in the experiment.

results of several consecutive frames in order to improve the system's accuracy. In other words, we will ignore the result that the accumulation frame number equals to 1 because it is not practical which will affect the decision of final selected region. However, too long of an accumulation time may have an adverse effect on the accuracy, since one object may appear in a short time. It is also important to note that the processing time increases with the rising of accumulation time.

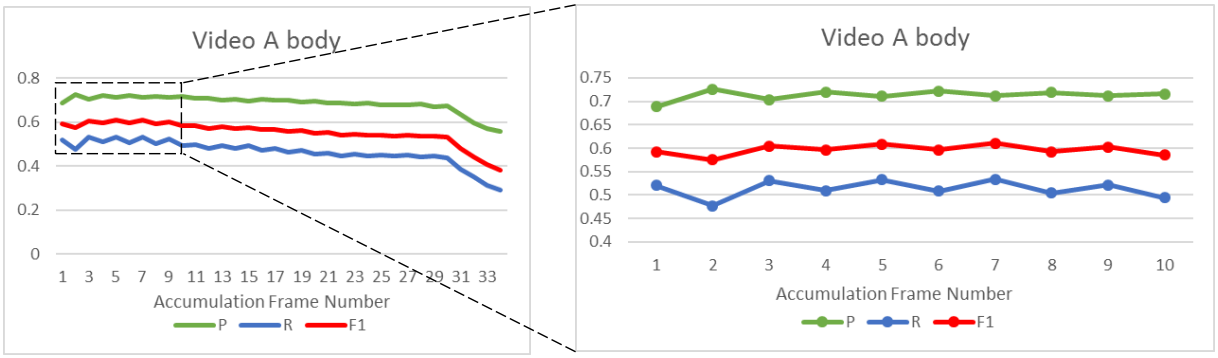
Although the accumulation time only occupies ω frames, which is a small part of the presentation, as shown in Figure 3.9, its processing time is much than the tracking frame time $T_{tracking}$.

As the goal of this experiment is to find the optimal accumulation time, we analyze the accuracy of detection using different accumulation times. Since the accumulation time increases in proportion to the processing time, we need to keep the accumulation time as smaller as possible, as long as the detection remains accurate.

Figures 5.2, 5.3, 5.4 and 5.4 show the results for foreground detection and human body detection of four sample videos A, B, C, D. The results are evaluated using three evaluation



(a) Forground Detection A



(b) Body detection A

Figure 5.2: Optimize accumulation time result of video A

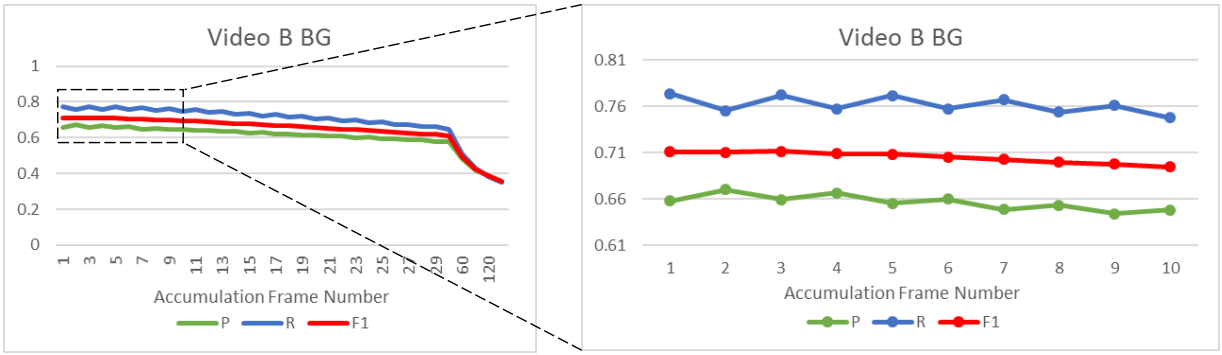
criteria:

$$precision = \frac{\#\ of\ detected\ pixels\ correctly\ identified\ by\ the\ detection\ method}{\#\ of\ detected\ pixels\ detected\ by\ the\ detection\ method} \tag{5.1a}$$

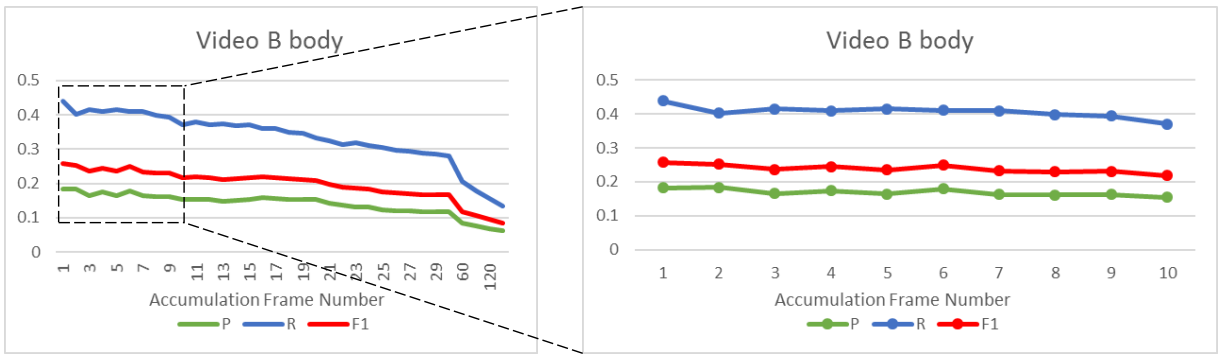
$$recall = \frac{\#\ of\ detected\ pixels\ correctly\ identified\ by\ the\ algorithm}{\#\ of\ detected\ pixels\ in\ ground-truth} \tag{5.1b}$$

$$F1\ score = \frac{2\ *\ precision*\ recall}{precision\ +\ recall} \tag{5.1c}$$

When applied to the entire sequence, the recall and precision reported are averages for all of the measured frames. The x-axis shows ω, which represents the number of accumulation frames, and the y-axis presents the value of each evaluation criteria for each



(a) Forground Detection B

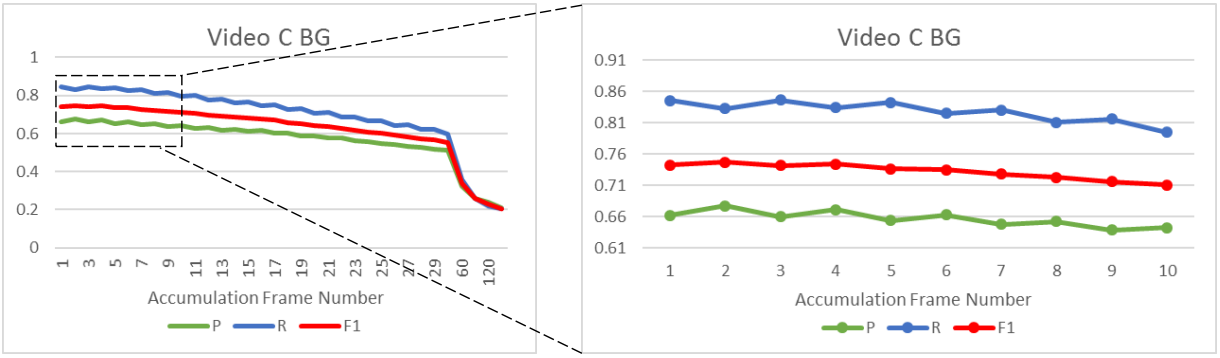


(b) Body detection B

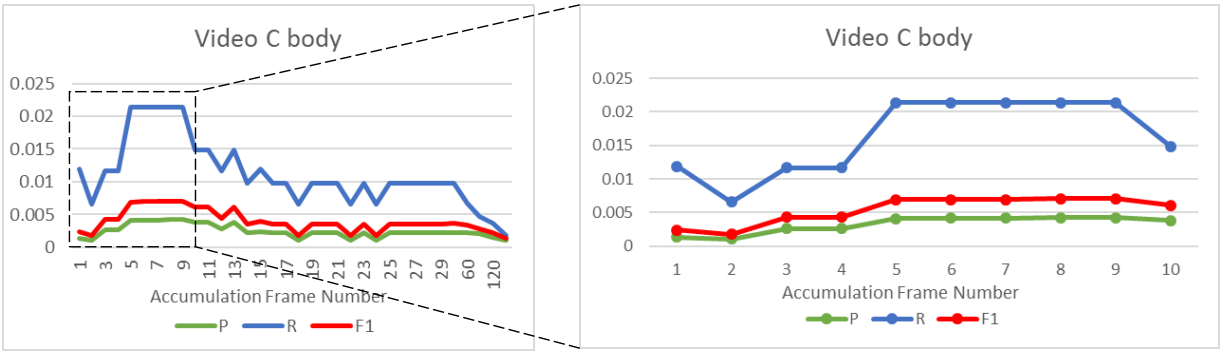
Figure 5.3: Optimize accumulation time result of video B

frame. The value is measured at the pixel level. There are three lines in the graph; the green line is the value of precision under different resolution values; the blue line is the recall; the red line is the F1 score. Typically, F1 score is a trade-off calculation between recall and precision. Recall usually increases with the number of pixels detected from different detection methods, which in turn may lead to a decrease in precision. A good detection algorithm should attain as high a precision value as possible without sacrificing recall. The ground truth is obtained by manually annotating the regions for each video with bounding boxes.

The results of video A are shown in Figure 5.2. In Figure 5.2(a), the graphs illustrate the results of the foreground detection method for Video A. We can see that the overall result is stable, while the performance slightly drops with increasing of the number of accumulated frames. The right sub-figure in Figure 5.2(a) further details the results for



(a) Forground Detection C



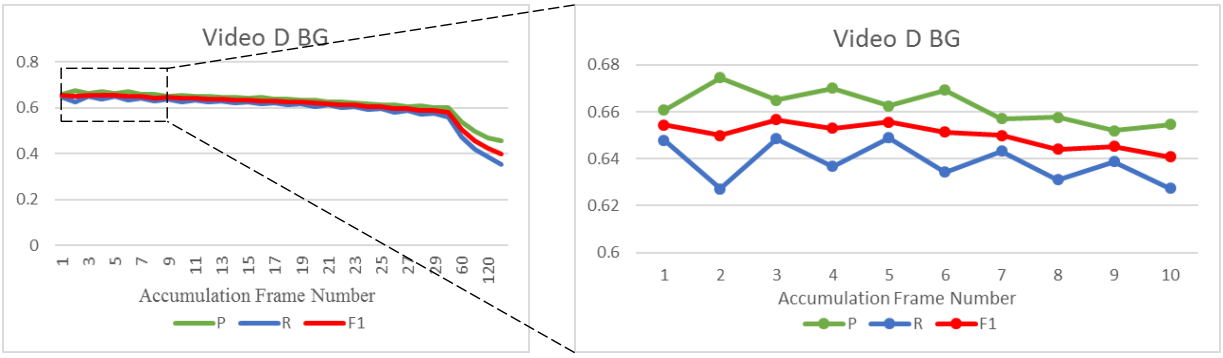
(b) Body detection C

Figure 5.4: Optimize accumulation time result of video C

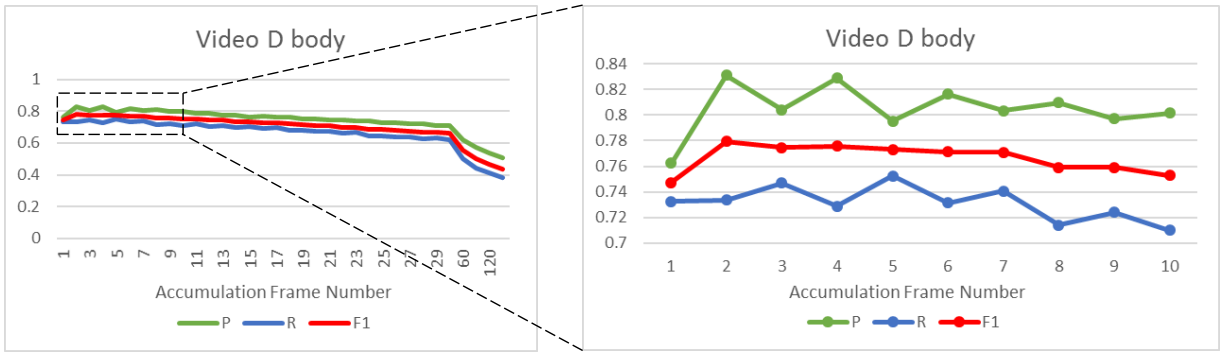
different values of x-axis ranging from 1 to 10. After observing the result, we realized that the value from 1 to 10 of x-axis part could be extracted. According to the F1 score, x-axis values $\{2, 3, 4\}$ are appropriate options, as they achieve better performances with respect to precision, recall and F1 score than. the other values. The Figure 5.2(b) shows the results from body detection. Similarly, the best accumulation time for the body detection of video A is between 2 to 7. Again, since the result is not sensitive, it is not very precise in determining which one is the best result, but the premise is to reduce the processing time as much as possible. In the system, after synthesizing these two results, $\omega = 2$ *or* 3 *or* 4 can be the optimized accumulation time for video A.

The results of video B are shown in Figure 5.3. Figure 5.3(a) and Figure 5.3(b) show the results of foreground detection and human body detection separately. Again, the performance trend of video B is similar as video A, the result range of ω could be from 2

to 4, which is the optimized accumulation time for video B.



(a) Forground Detection D



(b) Body detection D

Figure 5.5: Optimize accumulation time result of video D

For video C, its results are shown in Figure 5.4. Figure 5.4(a) and Figure 5.4(b) are corresponding to the foreground detection and body detection separately. For foreground detection result, similarly, we can get the range of ω could be from 2 to 4. However, the value of precision, recall and F1 score in human body detection is in a very low range, which means the body detection used in our system doesn't perform well under this video. Considering the scene in it, the human body is usually far away from the position of the camera, there are many missing bodies across the videos. Thus the performance is sensitive to the accumulation time. If ω is too small, there are not enough corrected bodies in the sensitivity map, which cannot be used for differentiating correct bodies with random noises. On the other hand, if ω is too large, the regions of bodies will be recognized as random noises as many bodies are not successfully detected. However, due to the fact that the bodies, which are far away from the camera position, are out of the effective range

of surveillance camera, they are relatively less important for the purpose of monitoring current camera. But foreground detection still works fine. So, the optimized resolution of body detection for video C could be set following with the result in foreground detection. The final range of ω could be also chosen from 2 to 4.

For video D, the results are shown in 5.5. Again, two detection methods are included. Figure 5.5(a) and 5.5(b) further shows the result similar as video A and B, so the result range of ω could be from 2 to 4.

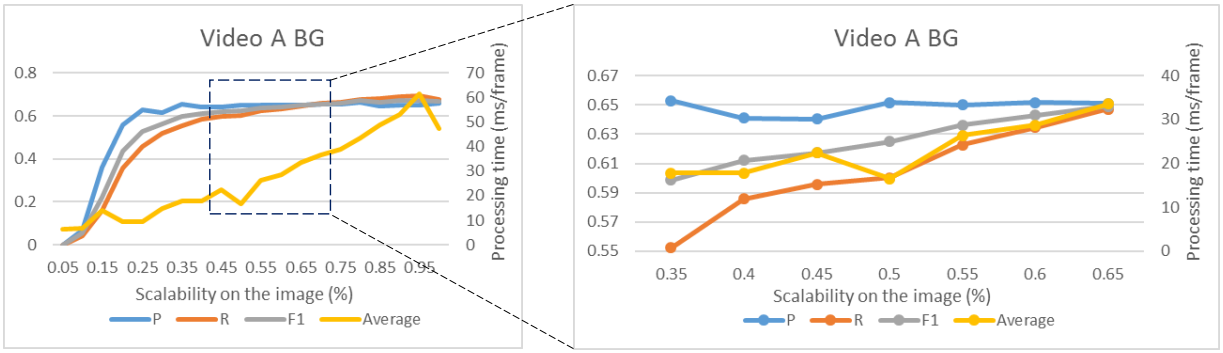
In brief, the algorithms is not very sensitive to the parameters setting. For different scenes, there is a range of choosing optimized result for both two detection methods without decreasing the performance. So, our experiment still illustrates that finding an optimized accumulation value helps reduce the effect of false detection.

5.2 Optimize Resolution Value

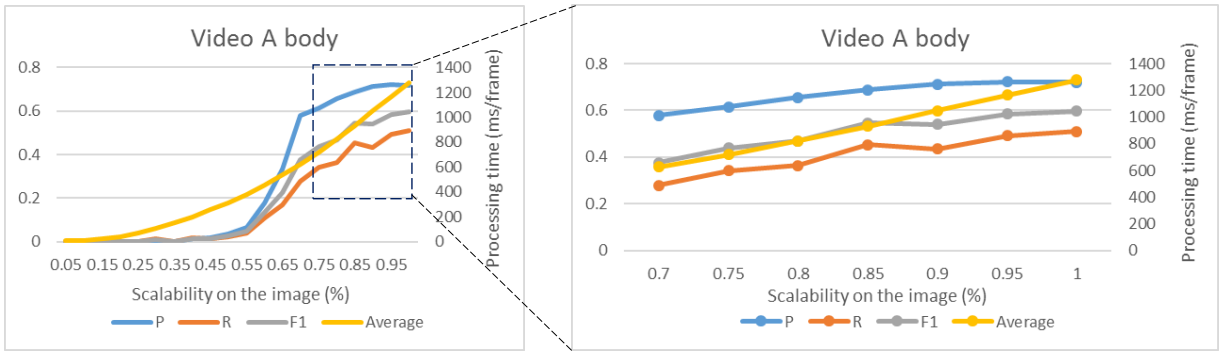
In this experiment, we vary the resolution of the image used for the detection methods and measure the accuracy. The goal is to reduce the resolution as much as possible, in order to reduce the processing time, without compromise the accuracy too much. The experiment results also include precision, recall and an F1 score. The ground truth is obtained in the same way as for the previous experiments. The accuracy is also measured at the pixel level.

Figure 5.6, Figure 5.7, Figure 5.8 and Figure 5.9 show the results of the experiments. The blue line is the precision value; the red line is the recall value; the gray line is the F1 score value, and the yellow line is the average processing time for each frame, under different resolution values. The x-axis shows the different resolution ratios of the image. The left y-axis shows the value of precision, recall and F1 score for the detection method using different resolution values. The right y-axis shows the average processing time in milliseconds per frame under different resolution values.

For video A, see Figure 5.6(a) and Figure 5.6(b). In Figure 5.6(a), the graph illustrates the results of the foreground detection method for video A. The performance is consistently



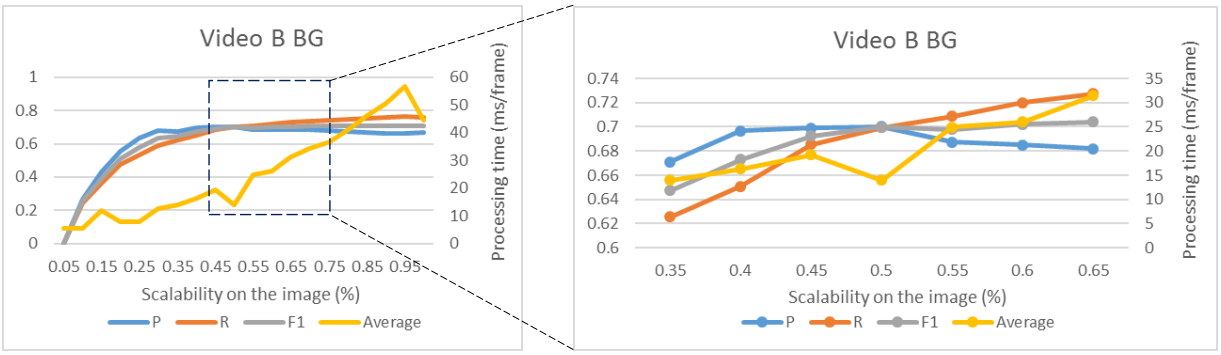
(a) Forground Detection A



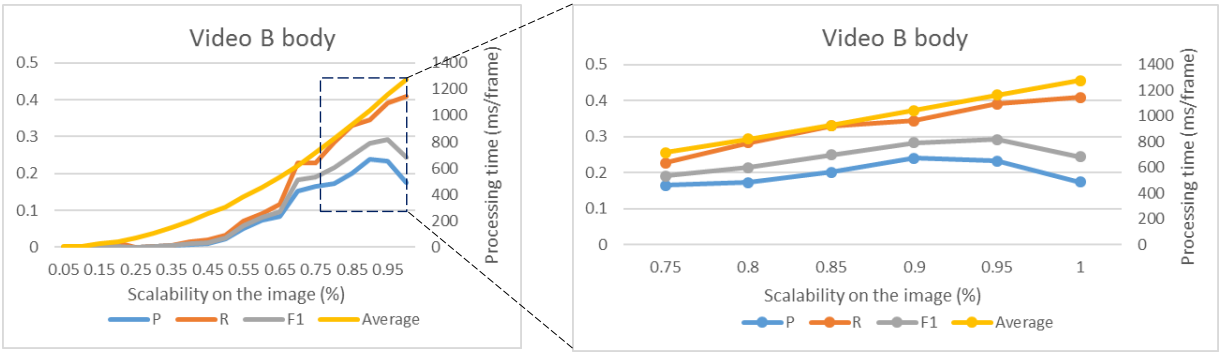
(b) Body detection A

Figure 5.6: Optimize resolution value result of video A

improved with the increasing of scaling ratio. Lower ratios will hurt the performance significantly as many details are missing after down-scaling. The overall trend of precision and recall stabilizes once the image resolution is over 0.35. Since the goal is to find an optimized resolution value for each detection method, we further detail the results corresponds to value ranging from 0.35 to 0.65 in the right sub-figure. We can see that when the resolution (x-axis) value equals 0.5 and the F1 score (left y-axis) equals to 0.63, the processing time is lower than the others. Thus, without compromising too much accuracy, the optimized resolution of the foreground detection method for video A is equal to 0.5. Similarly, the resolution value of 0.84 is the optimized value of the body detection method showed in Figure 5.6 (b) for video A. We can see that the optimal resolution ratio of body detection is higher than foreground detection. This is because that human body is relatively small in the surveillance videos, down-scaling may give negative impact on the detection performance.



(a) Forground Detection B



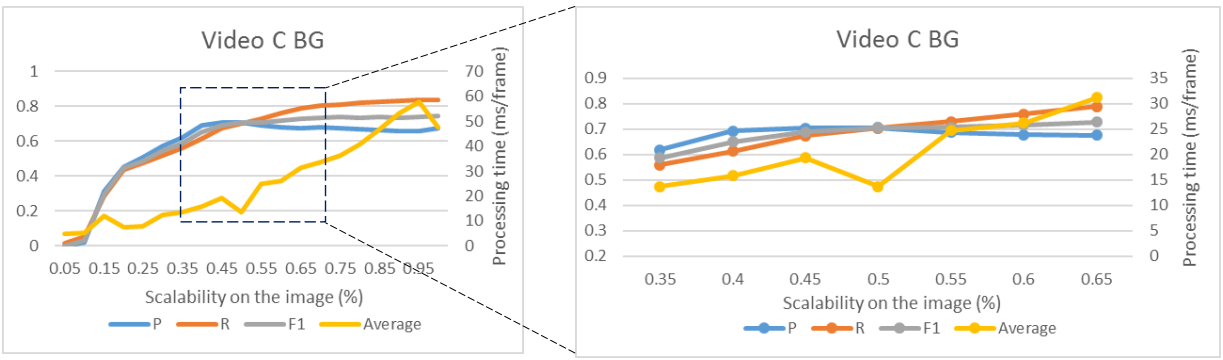
(b) Body detection B

Figure 5.7: Optimize resolution value result of video B

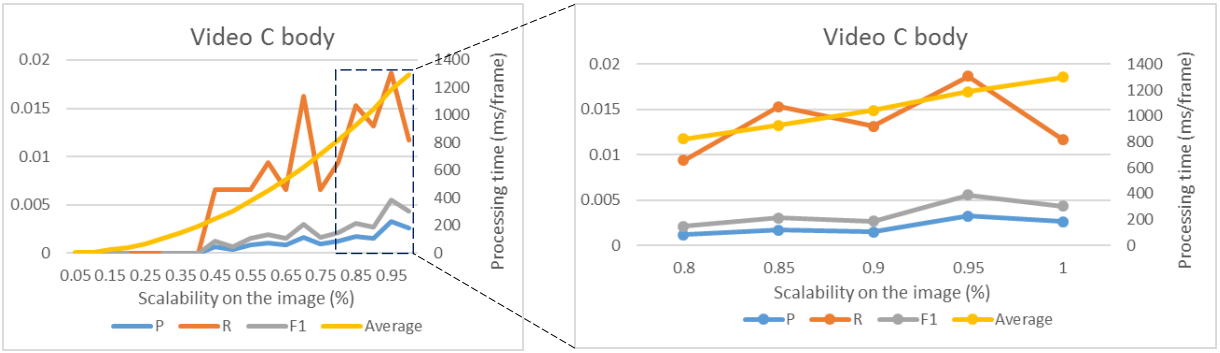
For video B, see Figure 5.7(a) and Figure 5.7(b). Similarly, for the foreground detection of video B, the optimized resolution value is equal to 0.5. For the body detection of video B, the optimized resolution is equal to 0.95. For video C, see Figure 5.8(a) and Figure 5.8(b). Similarly, for the foreground detection of video C, the optimized resolution value is equal to 0.5. Again we find the situation that was mentioned in the previous section, where the body detection method doesn't work very well, therefore the optimized resolution of body detection for video C could be 1. In other words, down-scaling is not suitable for this video.

For video D, see Figure 5.9(a) and Figure 5.9(b). Similarly, for the foreground detection of video D, the optimized resolution value is equal to 0.5. For the body detection of video D, the optimized resolution is equal to 0.8.

Note that the operation of scaling frames will introduce new computational costs. The



(a) Forground Detection C

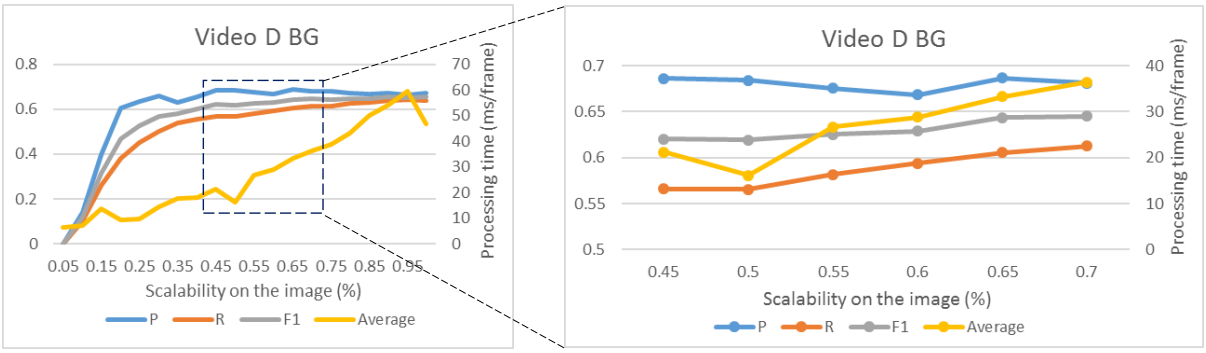


(b) Body detection C

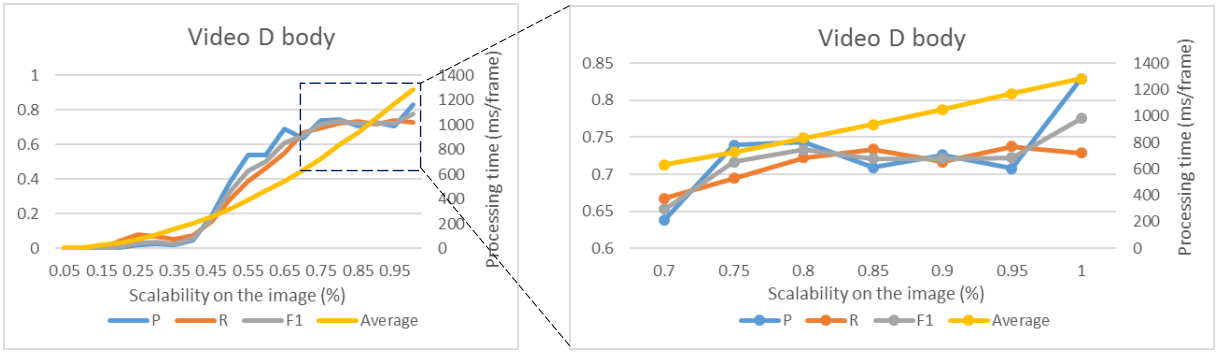
Figure 5.8: Optimize resolution value result of video C

time cost is determined by the scaling ratio. Rescaling an image by exactly a factor of 0.5 is computationally efficient, because it can be implemented by simply discarding every odd pixel. Since foreground detection is a very fast operation, the rescaling operation accounts for a large portion of the total time. This can be seen as a jump at x=0.5 in the plots. Body detection is much slower, so the additional time of scaling operation can be neglected, and no jump is visible.

In brief, although different scenes need different parameter settings in the system, our experiments still illustrate that finding an optimized resolution value can save processing time for each frame, without compromising the accuracy too much, which will also be proven in Table 5.3.



(a) Forground Detection D



(b) Body detection D

Figure 5.9: Optimize resolution value result of video D

5.3 Performance Comparison

In order to see how much we could improve accuracy with the proposed system compared to the simplified version as showed in Table 5.1, we conducted an accuracy comparison experiment. The way of calculating accuracy is

$$Accuracy = \frac{the\ times\ of\ zooming\ into\ the\ most\ inforamtive\ region}{the\ times\ of\ zooming\ into\ the\ most\ ground\ truth\ region} \tag{5.2}$$

and the ground truth is manually labeled based on the understanding of the video. Different from the experiments in previous sections, where ground-truth is generated by labeling regions of interest at pixel level, the accuracy defined in Equation 5.2 measures the overall precision of the proposed method on localizing the interesting regions. The two compared methods are simplified system and the proposed system. The final result is shown in Figure 5.10. The Y-axis represents the accuracy of the times of zoom into the correct regions. The

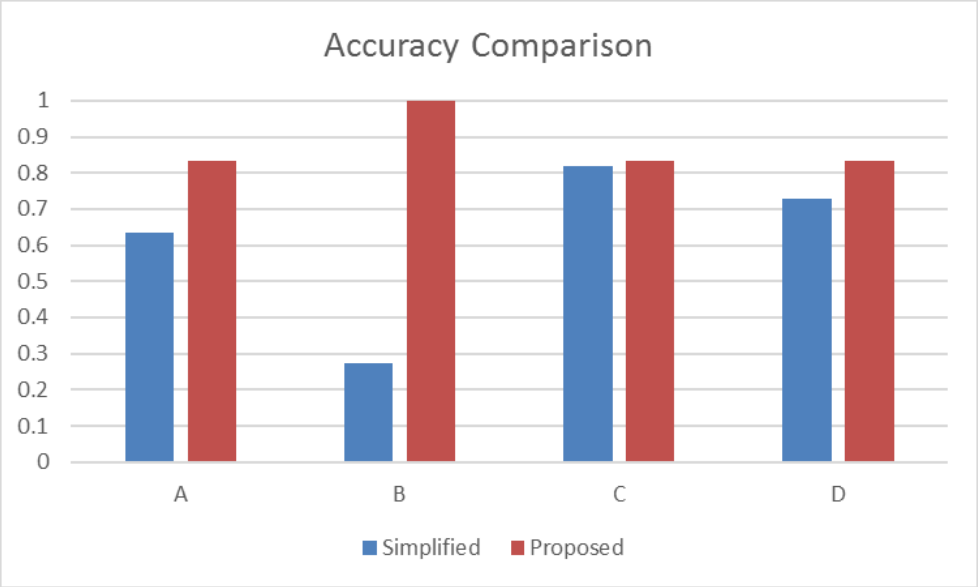


Figure 5.10: Accuracy comparison between simplified system and proposed system

main difference between these two methods are presentation frame $\triangle$. The $\triangle$ of simplified method equals to 150 frames. The $\triangle$ of proposed method equals to 300 frames.

According to the results in Figure 5.10, for all the sample videos, our proposed method performs better than the simplified method. We can see that for video B, the proposed method is 100% accuracy and the simplified method is below 30%. The reason is that in the site of video B, a small pillar is mis-detected as a person from start to end. The mask of this pillar is always appeared and accumulated on the sensitivity map, which finally affects the final display result. This shows one shortage of a long accumulation time in simplified system that once an meaningless object is mis-detected by a detection method, the sensitivity map will be going to worser increasingly.

In brief, the detection accuracy of our method is better than the simplified method even if we adopt a lower resolution for each frame. It shows that in our system, the accuracy does not lose, but increases a little bit as the randomly appeared noises are filtered by the proposed method.

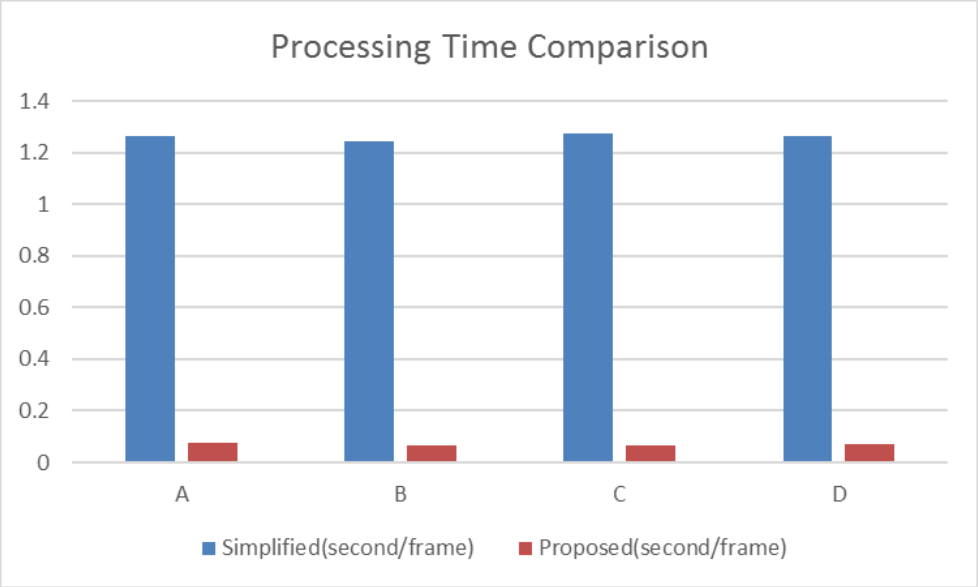


Figure 5.11: Processing time comparison between simplified system and proposed system

5.4 Processing Time Comparison

In order to see how much improvement on efficiency our proposed system could achieve, we calculated the average processing time for the four sample videos used above and then averaged over frames, which is shown in Figure 5.11. The two compared methods are the simplified system and the proposed system showed in Table 5.1. The Y-axis represents the average processing time of each frame in the video, whose unit is second per frame. The main difference between these two methods are accumulation time ω. The ω in simplified method is 150, but it is only 4 in our proposed system. In other words, ROI detection only functions on the proposed method over 4 frames, but does on the simplified method over 150 frames.

For all the sample videos, the average processing time of our proposed method is shorter than the simplified method. It is reasonable that our proposed method only accumulates sensitive map over several frames instead of doing over hundred frames. Meanwhile, the tracking method we used is also very efficient so that the average processing time on each frame could be much faster than the simplified system.

5.5 User Study

The main goal of our approach is to resize the video from a surveillance camera so that it can be used for remote monitoring on a mobile device with a small screen size. Considering that the proposed system contains multiple features, we perform two user studies on several systems with and without tracking listed in Table 5.1. This is because that the tracking mechanism will change the display style, by always putting the target in the center of the video. Comparing with the scaled and pan approach that zooms into or pans between selected regions in a fixed way is not quite objective. Thus, we first compare our simplified approach to the scaled and pan methods in a way of opinion based user study in section 5.5.1. Then, the proposed system using tracking is compared to the traditional scaled approach in section 5.5.2. The scaled system doesn't include any technology, which demonstrates the original and most common way of watching surveillance video. The pan system only considers the foreground detection and the human body detection, which are considered region detection. The display method of this system adapts the pan way, where the former selected region shifts to the next selected region, like a lens shifts from one point to another.

5.5.1 First User Study

In this part, we evaluated our approach with 22 male and 7 female non-expert subjects. The average age was 25, with a standard deviation of 4.

As the target resolution, we chose 384×216, which is 1/5 of the original resolution. Three different videos of the target resolution were created for each scenario:

- A scaled down version of the original video served as a baseline for the evaluation ("Scaled"). Selected images of Video B are shown in Figure 5.12.
- A video that uses the proposed ROI selection, but only pans between the ROIs without zooming out to the full overview in between ("Pan"). Selected images of Video B are shown in Figure 5.13.

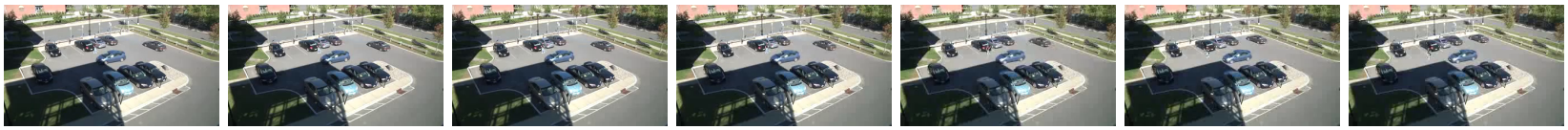


Figure 5.12: Results of Scaled method of Video B

Figure 5.13: Results of Pan method of Video B

Figure 5.14: Results of Simplified method of Video B

Figure 5.15: Results of Proposed method of Video B

Statement	Content
1	The video provides full coverage of the site.
2	The video provides all details of the site.
3	The camera motion was helpful in monitoring the area.
4	The video is boring.
5	It is easy to understand the activities in the video.

Table 5.2: Questionnaire of first user study

Statement	Scaled	Pan Only	Simplified
1	3.1 (1.4)	3.3 (1.0)	4.5 (0.9)
2	2.6 (1.2)	3.3 (0.8)	4.3 (1.0)
3	2.6 (1.2)	3.2 (1.0)	4.5 (0.8)
4	3.7 (1.0)	3.1 (1.0)	2.1 (1.2)
5	2.7 (1.3)	3.3 (1.0)	4.4 (0.8)

Table 5.3: First user study result

- The video created by the proposed simplified system ("Simplified"). Selected images of Video B are shown in Figure 5.14.

The users were given a brief introduction. Their task was to assume the role of a security operator who monitors the area and detects abnormal activities. All three versions of the scenario were shown next to each other simultaneously. After watching the videos, the users were given five statements about each version. They had to rate each statement shown in Table 5.2 on a scale from 1 (strongly disagree) to 5 (strongly agree). The website we used for the study can be found at [1].

Table 5.3 shows the results of the study, including the agreement scores for the five

[1] https://sites.google.com/site/acm2014ssa/

statements and the three approaches. The values are averaged over the four scenes and all participants, with the standard deviation given in parentheses. Since the users' ratings were similar across all four scenarios, we only show the averaged values here. From the table, it can be seen that the simplified approach achieves better results than the other two methods, in all of the five categories considered. Note that for statement 4 (the video is boring), a lower value is better. For the two other approaches (scaled and pan), the video that pans between the ROIs seems to be more useful. This is due to the small output resolution. The scaled version was too small to be useful for monitoring. In statements 1, 3 and 5, our approach obtained an average score between 4.3 and 4.5. This indicates that the participants of the study generally found our retargeting approach to provide videos that were helpful in the given surveillance task.

5.5.2 Second User Study

The objectives of the second user study are to prove that:

1. The output video created by our system provides a more detailed view of the surveillance site in comparison to other videos.
2. The details are provided in such a way that the user does not loose the context required for situation assessment.
3. The system zooms into the regions that are relevant from a surveillance perspective.
4. The video is engaging, i.e., the operator's attention level remains higher while viewing our video in comparison to other videos.

We conducted two different types of user studies to establish these points: opinion based user study and activity based user study. In the first part, we record the opinions of users about specific questions as ratings, which is similar to the way used in section 5.5.1. To differentiate with the study in section 5.5.1, we slightly modify the statements showed in table 5.4 to acquire more specific opinions from users. In the second type of user study, we assign specific tasks to the users, such as detecting activities, and measure detection

accuracy with a questionnaire. In other words, we compare the amount of information that a user can get from a sample video, which is one of the videos used in the first user study. The only two differences between the simplified and the proposed systems are that the simplified system lacks tracking and two optimization methods. The experiments that prove the optimization were done in Sections 5.1 and 5.2, and show that the proposed system doesn't lose too much accuracy when detecting informative regions. Thus, the simplified method will not be included in this user study.

We evaluated our approach in an opinion based user study with 24 participants (19 male and 7 female), none of which were experts on the subject. The average age was 25 with a standard deviation of 4. They participated in evaluating the following videos: Proposed A, Scaled A, Proposed B, Scaled B, Proposed C, and Scaled D. In our activity based user study, there were also 24 participants (17 male, 7 female) invited, none of which were experts on the subject. The average age was 25 with a standard deviation of 3.8. The test videos they used were: Proposed A, Scaled A, Proposed B, Scaled B, Scaled C, and Proposed D. Videos A and B are assigned in the opinion based user study and Videos C and D are in another study. Considering that the activity based user study is an essay style question, we switched the proposed method and scaled videos C and D. The reason is very simple. Once the video has been watched, watching it again will affect the participants answer, since they already know the site; we therefore divided these two videos.

The website of the user study can be found at [2] [3].

Opinion Based User Study - 1

In this experiment, we ask the users to play the role of a security operator whose main task is to monitor a surveillance site and observe important activities in order to detect any abnormal situations. The users watch two sets of videos from videos A and B, separately, and rate the five statements shown in Table 5.4 on a scale of 1 to 5:

Table 5.5 shows the results of the study, which include the agreement scores for the five

[2] https://sites.google.com/site/userstudytype1/

[3] https://sites.google.com/site/userstudytype2/

Statement	Content
1	The video is able to provide details of the regions that are important for surveillance.
2	The video provides detailed view of all important events and activities happening at the site.
3	The camera operations reduce boredom and keep me engaged to the activities in the video.
4	The camera operations are helpful in monitoring the area.
5	It is easy to understand the activities in the video.

Table 5.4: Questionnaire of opinion based user study - 1

Statement	Scaled A	Proposed A	Scaled B	Proposed B
1	2.8 (1.1)	4.3 (1.0)	2.9 (1.0)	4.5 (0.7)
2	3.0 (1.1)	4.2 (0.9)	2.8 (1.0)	4.3 (0.80)
3	2.8 (1.0)	4.1 (1.0)	2.70 (1.0)	4.3 (0.7)
4	3.2 (1.0)	4.2 (1.0)	3.0 (0.9)	4.2 (0.8)
5	2.9 (0.8)	4.2 (0.9)	3.1 (1.0)	4.2 (0.8)

Table 5.5: Opinion based user study result

statements and the two approaches. The values are averaged for all participants, with the standard deviation given in parentheses. From the table, it can be seen that the proposed approach achieves better results than the scaled one. The results of two scaled videos are also similar as we did in Table 5.3.

Activity Based User Study - 2

For this experiment, according to the activity that took place in videos C and D, we asked 3 questions for video C and 6 questions for video D concerning the detailed view of the site, which are shown in Table 5.6.

Table 5.7 shows the results of the study, including the agreement scores for the state-

Video	Statement	Content
C	1	What is the person by the car doing in the beginning (first 10 s) of the video?
	2	What is the the person with the dolly doing? Dolly is a small platform on wheels used for holding heavy objects such as a TV, desktop, etc.
	3	How many people does the largest group of people consist of? People tanding close to each other form a group.
D	1	What is the right person of the group of two in the beginning of the video holding in his hand?
	2	How many people in the video are wearing a hat?
	3	What does the person in the red shirt waiting by the posts do when he sees his friend arriving by car?
	4	What is the person in the black shirt doing while getting out of the car?
	5	How many groups of people are walking through the video?
	6	Where are the two people (purple shirt and orange/purple striped shirt) before they walked through the scene?

Table 5.6: Questionnaire of activity based user study - 2 for video C and D

Statement	Scaled C	Proposed C	Scaled D	Proposed D
1	0.22	0.77	0.08	0.47
2	0.33	0.65	0.31	0.58
3	0.5	0.88	0.61	0.68
4	N/A	N/A	0.30	0.68
5	N/A	N/A	0.46	0.68
6	N/A	N/A	0.46	0.79

Table 5.7: Activity based user study result

ments and the two approaches. The values are averaged for all of the participants. From the table, it can be seen that the proposed approach achieves better results than the scaled one, especially when it comes to the details of the scene. Statement 1 of video D asks about a detailed view of a person holding a cup, which is really difficult to spot with the scaled

method, so that an accuracy score of 0.08 is reasonable. However, the proposed method got a 0.47 accuracy score, even though the cup is a tiny object. Similarly, the answer of statement 1 for video C is that a person is making a phone call to someone. Statement 1 for the Scaled C video only has an accuracy of 0.22. On the contrary, the proposed method achieves a 0.77 accuracy score, which shows that our method is effective when watching details of the video.

In short, we proposed four expectations before doing this user study. The results are reasonable and prove that the proposed method provides a more detailed view of the surveillance scene and is helpful to engage the people who are watching the video.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In this work, we proposed a smart assistant for remote video surveillance on a handheld device. A high-resolution surveillance video was retargeted to a smaller resolution to be displayable on a small screen and save bandwidth. This was done by first identifying regions of interest in the video and then tracking the selected object and zooming into specified regions one after another. An overview of the entire screen is given periodically, to provide the necessary context to the operator. Our proposed presentation method improves the effectiveness of operators doing surveillance task through handheld devices. The proposed learning method makes the output video more natural and comfortable to view. The calculation average result of optimizing accumulation times is 4. The average optimized resolution values of foreground detection and human body detection are 0.5 and 0.8 individually. The total processing time has been improved significantly which is averagely equal to 0.08 second fer frame. The user study is separated into two parts for a better comparison. The results of these two user studies show that the proposed method provides full coverage of the scene while also showing the amount of detail necessary for an understanding of the activities therein. The users generally reported that the smart assistant system was helpful in remote monitoring a scene under surveillance.

6.2 Future work

The proposed system is compatible with a single surveillance camera of HD quality. However, for the system to be widely used, an increase in the number of surveillance cameras is required. The multi-angle surveillance camera setting is helpful to monitor more detailed and complete scenes around the protected object. Providing an effective layout where multiple videos are shown on a single screen of a smart device is challenging work, as is picking which monitored scenes are more informative.

Despite the fact that each detection method used in the system was optimized at the resolution level, the processing speed can still be accelerated. Nowadays, a lot of new adaptive algorithms for foreground detection and human body detection are more reliable. Updating detection methods could also be meaningful work. As several parameters such as the thresholds in the algorithms that detect objects are manually settled down, an adaptive threshold setting is worthy of further study.

Realizing an automated surveillance system is feasible as well, including supporting activity recognition, danger alarms, etc. The aim is to increase security and safety in various application domains such as traffic monitoring, yard protection, etc.

References

[1] Shai Avidan. Support vector tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(8):1064–1072, 2004.

[2] Yannick Benezeth, Pierre-Marc Jodoin, Bruno Emile, Hélene Laurent, and Christophe Rosenberger. Comparative study of background subtraction algorithms. *Journal of Electronic Imaging*, 19(3):033003–033003, 2010.

[3] Marcelo Bertalmio, Guillermo Sapiro, and Gregory Randall. Morphing active contours. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(7):733–737, 2000.

[4] Michael J Black and Allan D Jepson. Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. *International Journal of Computer Vision*, 26(1):63–84, 1998.

[5] Andrew Blake, Michael Isard, et al. *Active contours: the application of techniques from graphics, vision, control theory and statistics to visual tracking of shapes in motion*, volume 1. Springer London, 2000.

[6] TE Boult, R Micheals, X Gao, P Lewis, C Power, W Yin, and A Erkan. Frame-rate omnidirectional surveillance and tracking of camouflaged and occluded targets. In *Proceedings Second IEEE Workshop on Visual Surveillance*, pages 48–55. IEEE, 1999.

[7] Ted J Broida and Rama Chellappa. Estimation of object motion parameters from noisy images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (1):90–99, 1986.

[8] Sebastian Brutzer, Benjamin Hoferlin, and Gunther Heidemann. Evaluation of background subtraction techniques for video surveillance. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1937–1944. IEEE, 2011.

[9] Christopher JC Burges. A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, 2(2):121–167, 1998.

[10] Simone Calderara, Rudy Melli, Andrea Prati, and Rita Cucchiara. Reliable background suppression for complex scenes. In *Proceedings of the 4th ACM international workshop on Video surveillance and sensor networks*, pages 211–214. ACM, 2006.

[11] Sen-Ching S Cheung and Chandrika Kamath. Robust techniques for background subtraction in urban traffic video. In *Proceedings of SPIE*, volume 5308, pages 881–892, 2004.

[12] Won-Ho Chung. A smartphone watch for mobile surveillance service. *Personal and Ubiquitous Computing*, 16(6):687–696, 2012.

[13] Dorin Comaniciu. Bayesian kernel tracking. In *Pattern Recognition*, pages 438–445. Springer, 2002.

[14] Dorin Comaniciu, Visvanathan Ramesh, and Peter Meer. Real-time tracking of non-rigid objects using mean shift. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 142–149. IEEE, 2000.

[15] Nello Cristianini and John Shawe-Taylor. An introduction to support vector machines, 2000.

[16] Rita Cucchiara, Costantino Grana, Massimo Piccardi, and Andrea Prati. Detecting moving objects, ghosts, and shadows in video streams. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(10):1337–1342, 2003.

[17] Rita Cucchiara and Giovanni Gualdi. Mobile video surveillance systems: An architectural overview. In *Mobile Multimedia Processing*, pages 89–109. Springer, 2010.

[18] Ross Cutler and Larry Davis. View-based detection and analysis of periodic motion. In *Proceedings of International Conference on Pattern Recognition*, volume 1, pages 495–495. IEEE Computer Society, 1998.

[19] Ross Cutler and Larry S. Davis. Robust real-time periodic motion detection, analysis, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):781–796, 2000.

[20] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 886–893. IEEE, 2005.

[21] Thomas Deselaers, Philippe Dreuw, and Hermann Ney. Pan, zoom, scantime-coherent, trained automatic video cropping. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.

[22] Piotr Dollar, Christian Wojek, Bernt Schiele, and Pietro Perona. Pedestrian detection: An evaluation of the state of the art. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(4):743–761, 2012.

[23] Hazem El-Alfy, David Jacobs, and Larry Davis. Multi-scale video cropping. In *Proceedings of international conference on Multimedia*, pages 97–106. ACM, 2007.

[24] Ahmed Elgammal, David Harwood, and Larry Davis. Non-parametric model for background subtraction. In *Computer VisionECCV 2000*, pages 751–767. Springer, 2000.

[25] Markus Enzweiler and Dariu M Gavrila. Monocular pedestrian detection: Survey and experiments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12):2179–2195, 2009.

[26] David A Forsyth and Jean Ponce. *Computer Vision: A Modern Approach*. Prentice Hall Professional Technical Reference, 2002.

[27] Nir Friedman and Stuart Russell. Image segmentation in video sequences: A probabilistic approach. In *Proceedings of the Thirteenth conference on Uncertainty in artificial intelligence*, pages 175–181. Morgan Kaufmann Publishers Inc., 1997.

[28] Hironobu Fujiyoshi, Alan J Lipton, and Takeo Kanade. Real-time human motion analysis by image skeletonization. *IEICE TRANSACTIONS on Information and Systems*, 87(1):113–120, 2004.

[29] Keinosuke Fukunaga and Larry Hostetler. The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Transactions on Information Theory*, 21(1):32–40, 1975.

[30] Roberto Gallea, Edoardo Ardizzone, and Roberto Pirrone. Physical metaphor for streaming media retargeting. *IEEE transactions on multimedia*, 16(4):971–979, 2014.

[31] Brian Gloyer, Hamid K Aghajan, Kai-Yeung Siu, and Thomas Kailath. Video-based freeway-monitoring system using recursive vehicle tracking. In *Proceedings of IS&T/SPIE's Symposium on Electronic Imaging: Science & Technology*, pages 173–180. International Society for Optics and Photonics, 1995.

[32] Matthias Grundmann, Vivek Kwatra, Mei Han, and Irfan Essa. Discontinuous seam-carving for video retargeting. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2010.

[33] Ismail Haritaoglu, David Harwood, and Larry S. Davis. W 4: Real-time surveillance of people and their activities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):809–830, 2000.

[34] Janne Heikkilä and Olli Silvén. A real-time system for monitoring of cyclists and pedestrians. *Proceedings Second IEEE Workshop on Visual Surveillance*, 22(7):563–570, 2004.

[35] Weiming Hu, Tieniu Tan, Liang Wang, and Steve Maybank. A survey on visual surveillance of object motion and behaviors. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 34(3):334–352, 2004.

[36] Daniel P Huttenlocher, Jae J Noh, and William J Rucklidge. Tracking non-rigid objects in complex scenes. In *Proceedings of International Conference on Computer Vision*, pages 93–101. IEEE, 1993.

[37] Jinman Kang, Isaac Cohen, and Gerard Medioni. Object reacquisition using invariant appearance model. In *Proceedings of the 17th International Conference on Pattern Recognition*, volume 4, pages 759–762. IEEE, 2004.

[38] Hendrik Knoche, Marco Papaleo, M Angela Sasse, and Alessandro Vanelli-Coralli. The kindest cut: enhancing the user experience of mobile tv through adequate zooming. In *Proceedings of international conference on Multimedia*, pages 87–96. ACM, 2007.

[39] Stephan Kopf, Thomas Haenselmann, Johannes Kiess, Benjamin Guthier, and Wolfgang Effelsberg. Algorithms for video retargeting. *Multimedia Tools and Applications*, 51(2):819–861, 2011.

[40] Stephan Kopf, Johannes Kiess, Hendrik Lemelson, and Wolfgang Effelsberg. Fscav: fast seam carving for size adaptation of videos. In *Proceedings of ACM international conference on Multimedia*, pages 321–330. ACM, 2009.

[41] Chun-Ming Li, Yu-Shan Li, Shu-Hai Wang, and Xiu-Qing Zhang. Moving human body detection in video sequences. In *Proceedings of International Conference onMachine Learning and Cybernetics*, volume 4, pages 2188–2192. IEEE, 2007.

[42] Sheng-Tun Li, Huang-Chih Hsieh, Ly-Yen Shue, and Wen-Shen Chen. Pda watch for mobile surveillance services. In *Proceedings of IEEE Workshop on Knowledge Media Networking*, pages 49–54. IEEE, 2002.

[43] Yuanning Li, Yonghong Tian, Jingjing Yang, Ling-Yu Duan, and Wen Gao. Video retargeting with multi-scale trajectory optimization. In *Proceedings of the international conference on Multimedia information retrieval*, pages 45–54. ACM, 2010.

[44] Alan J Lipton, Hironobu Fujiyoshi, and Raju S Patil. Moving target classification and tracking from real-time video. In *Proceedings of IEEE Workshop on Applications of Computer Vision*, pages 8–14. IEEE, 1998.

[45] Fang Liu and Rosalind W Picard. Finding periodicity in space and time. In *Proceedings of International Conference on Computer Vision*, pages 376–383. IEEE, 1998.

[46] Feng Liu and Michael Gleicher. Video retargeting: automating pan and scan. In *Proceedings of annual ACM international conference on Multimedia*, pages 241–250. ACM, 2006.

[47] BPL Lo and SA Velastin. Automatic congestion detection system for underground platforms. In *Proceedings of 2001 International Symposium on Intelligent Multimedia, Video and Speech Processing*, pages 158–161. IEEE, 2001.

[48] Stephane G Mallat. A theory for multiresolution signal decomposition: the wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7):674–693, 1989.

[49] Nigel JB McFarlane and C Paddy Schofield. Segmentation and tracking of piglets in images. *British Machine Vision Conference*, 8(3):187–193, 1995.

[50] Christian Micheloni, Gian Luca Foresti, and Lauro Snidaro. A network of co-operative cameras for visual surveillance. In *IEE Proceedings- Vision, Image and Signal Processing*, volume 152, pages 205–212. IET, 2005.

[51] Davide A Migliore, Matteo Matteucci, and Matteo Naccari. A revaluation of frame difference in fast and robust motion detection. In *Proceedings of ACM international workshop on Video surveillance and sensor networks*, pages 215–218. ACM, 2006.

[52] Baback Moghaddam and Ming-Hsuan Yang. Sex with support vector machines. In *Advances in Neural Information Processing Systems*, pages 960–966. Citeseer, 2000.

[53] Neeti A Ogale. A survey of techniques for human detection from video. *Survey, University of Maryland*, 2006.

[54] Sangmin Oh, Anthony Hoogs, Amitha Perera, Naresh Cuntoor, Chia-Chih Chen, Jong Taek Lee, Saurajit Mukherjee, et al. A large-scale benchmark dataset for event recognition in surveillance video. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 3153–3160. IEEE, 2011.

[55] Michael Oren, Constantine Papageorgiou, Pawan Sinha, Edgar Osuna, and Tomaso Poggio. Pedestrian detection using wavelet templates. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 193–199. IEEE, 1997.

[56] Edgar Osuna, Robert Freund, and Federico Girosi. Training support vector machines: an application to face detection. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 130–136. IEEE, 1997.

[57] Donovan H Parks and Sidney S Fels. Evaluation of background subtraction algorithms with post-processing. In *Proceedings of IEEE Fifth International Conference on Advanced Video and Signal Based Surveillance*, pages 192–199. IEEE, 2008.

[58] Ramprasad Polana and Randal Nelson. Low level recognition of human motion (or how to get your man without finding his body parts). In *Proceedings of IEEE Workshop on Motion of Non-Rigid and Articulated Objects*, pages 77–82. IEEE, 1994.

[59] Ramprasad Polana and Randal C Nelson. Detection and recognition of periodic, nonrigid motion. *International Journal of Computer Vision*, 23(3):261–282, 1997.

[60] Ronald Poppe. A survey on vision-based human action recognition. *Image and vision computing*, 28(6):976–990, 2010.

[61] T Raty, Lassi Lehikoinen, and Francois Bremond. Scalable video transmission for a surveillance system. In *Proceedings of International Conference on Information Technology: New Generations*, pages 1011–1016. IEEE, 2008.

[62] Collins RetaI. A system for video surveillance and monitoring: Vsam final report. *Carnegie Melhm University: Technical Repo˜ CM U, t) ittsbmgh USA*, 2000.

[63] V Salari and Ishwar K. Sethi. Feature point correspondence in the presence of occlusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(1):87–91, 1990.

[64] Koichi Sato and Jake K Aggarwal. Temporal spatio-velocity transform and its application to tracking and interaction. *Computer Vision and Image Understanding*, 96(2):100–128, 2004.

[65] Ishwar K Sethi and Ramesh Jain. Finding trajectories of feature points in a monocular image sequence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (1):56–73, 1987.

[66] Denis Simakov, Yaron Caspi, Eli Shechtman, and Michal Irani. Summarizing visual data using bidirectional similarity. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.

[67] Chris Stauffer and W Eric L Grimson. Adaptive background mixture models for real-time tracking. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2. IEEE, 1999.

[68] Olivier Steiger, Touradj Ebrahimi, and Andrea Cavallaro. Surveillance video for mobile devices. In *Proceedings of IEEE Conference onAdvanced Video and Signal Based Surveillance*, pages 620–625. IEEE, 2005.

[69] Satoshi Suzuki et al. Topological structural analysis of digitized binary images by border following. *Computer Vision, Graphics, and Image Processing*, 30(1):32–46, 1985.

[70] Keigo Takahara, Takashi Toriu, and Thi Thi Zin. Making background subtraction robust to various illumination changes. *International Journal of Computer Science and Network Security*, 11(3):241, 2011.

[71] Kentaro Toyama, John Krumm, Barry Brumitt, and Brian Meyers. Wallflower: Principles and practice of background maintenance. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 1, pages 255–261. IEEE, 1999.

[72] Akira Utsumi and Nobuji Tetsutani. Human detection using geometrical pixel value structures. In *Proceedings of IEEE International Conference on Automatic Face and Gesture Recognition*, pages 34–39. IEEE, 2002.

[73] Cor J Veenman, Marcel JT Reinders, and Eric Backer. Resolving motion correspondence for densely moving points. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(1):54–72, 2001.

[74] Paul Viola, Michael J Jones, and Daniel Snow. Detecting pedestrians using patterns of motion and appearance. In *Proceedings of IEEE International Conference on Computer Vision*, pages 734–741. IEEE, 2003.

[75] Yu-Shuen Wang, Jen-Hung Hsiao, Olga Sorkine, and Tong-Yee Lee. Scalable and coherent video resizing with per-frame optimization. In *ACM Transactions on Graphics (TOG)*, volume 30, page 88. ACM, 2011.

[76] Yuan-Kai Wang, Li-Ya Wang, and Yung-Hsiang Hu. A mobile video surveillance system with intelligent object analysis. In *Electronic Imaging*, pages 68210I–68210I. International Society for Optics and Photonics, 2008.

[77] Christopher Richard Wren, Ali Azarbayejani, Trevor Darrell, and Alex Paul Pentland. Pfinder: Real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):780–785, 1997.

[78] Yang-Yang Xiang and Mohan S Kankanhalli. Video retargeting for aesthetic enhancement. In *Proceedings of the international conference on Multimedia*, pages 919–922. ACM, 2010.

[79] Alper Yilmaz, Omar Javed, and Mubarak Shah. Object tracking: A survey. *Acm computing surveys (CSUR)*, 38(4):13, 2006.

[80] Masayuki Yokoyama and Tomaso Poggio. A contour-based moving object detection and tracking. In *Proceedings of Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pages 271–276. IEEE, 2005.

[81] S Zhang, SC Chan, RD Qiu, KT Ng, YS Hung, and W Lu. On the design and implementation of a high definition multi-view intelligent video surveillance system. In

Proceedings of IEEE International Conference on Signal Processing, Communication and Computing (ICSPCC), pages 353–357. IEEE, 2012.

[82] Quming Zhou and Jake K Aggarwal. Tracking and classifying moving objects from video. In *Proceedings of IEEE Workshop on Performance Evaluation of Tracking and Surveillance*. Hawaii, USA, 2001.

[83] Dennis Zill, Warren S Wright, and Michael R Cullen. *Advanced engineering mathematics*. Jones & Bartlett Learning, 2011.

[84] Zoran Zivkovic. Improved adaptive gaussian mixture model for background subtraction. In *Proceedings of International Conference on Pattern Recognition*, volume 2, pages 28–31. IEEE, 2004.

[85] Zoran Zivkovic and Ferdinand van der Heijden. Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern recognition letters*, 27(7):773–780, 2006.