PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA
Ministry of Higher Education and Scientific Research
University of Amar Telidji - Laghouat

Faculty of Technology
Department of Electrical Engineering

PhD Thesis in Electrical Engineering

SEHAIRI Kamal

Design and Implementation of a Real-Time Intelligent Video Surveillance System

Conception et Implémentation en Temps Réel d'un Système de Vidéosurveillance Intelligent

Mr. Bouzouad Mouloud	Chair	Professor	U.A.T Laghouat, Algeria
Mrs. Chouireb Fatima	Supervisor	Professor	U.A.T Laghouat, Algeria
Mr. Meunier Jean	Co-supervisor	Professor	UdeM, Montreal, Canada
Mr. Larbes Cherif	Examiner	Professor	E.N.P, Algiers, Algeria
Mr. Djafer Djelloul	Examiner	Research Director	URAER, Ghardaïa, Algeria
Mr. Kious Mecheri	Examiner	Associate Professor	U.A.T Laghouat, Algeria

2017-2018

Abstract- This thesis presents and proposes several techniques for developing real time intelligent video surveillance systems ranging from simple image processing to the most sophisticated machine learning methods. The aim is to detect and recognize an abnormal situation in a video; in order to achieve that this work is split into three main parts. The first part consists of benchmarking different segmentations and motion detection methods. The second one consists of implementing a real time system for action recognition on hardware circuits, and the last part on developing high efficient system for recognizing fall of older people by proposing a new algorithm for extracting features.

The two originalities of this work are the development of this algorithm and the elaboration of a complete benchmark for motion detection methods.

Key words: Video processing, Behavior change detection, FPGA, Real time implementation.

ملخص: يقدم هذا التقرير ويقترح العديد من التقنيات لتطوير أنظمة المراقبة الذكية بالفيديو في الوقت الآني بدءا من معالجة الصور البسيطة إلى أساليب التعلم الآلي الأكثر تطورا، والهدف هو الكشف والتعرف على حالات غير طبيعية مشتبهة عبر الفيديو، من أجل تحقيق ذلك تم تقسيم هذا العمل البحثي إلى 3 أجزاء رئيسية، الجزء الأول يتألف من مقارنة مختلف لوغالايتمات تجزئة الصور وأساليب الكشف عن الحركة، الجزء الثاني يتمثل في تنفيذ نظام في الوقت الاني على دوائر خاصة بحيث يتمكن من معرفة عمل الاشخاص، الجزء الأخير يتمثل في تطوير نظام ذو كفاءة عالية للتعرف على حالات السقوط لدى كبار السن من خلال اقتراح خوارزمية جديدة لاستخراج الميزات.

وتتمثل أصالة هذا العمل في تطوير هذه الخوارزمية ووضع معيار كامل لأساليب الكشف عن الحركة.

كلمات مفتاحية: معالجة الفيديو، رصد تغير السلوك، FPGA ، التنصيب الآني.

Publication

Journals

- K. Sehairi, C. Benbouchama, and F. Chouireb, "Real Time Implementation on FPGA of Moving Objects Detection and Classification," International Journal of Circuits, Systems and Signal Processing, vol.9, pp 160-167, 2015.

- Kamal Sehairi, Fatima Chouireb, Jean Meunier, "Comparative study of motion detection methods for video surveillance systems," J. Electron. Imaging 26(2), 023025 (2017), doi: 10.1117/1.JEI.26.2.023025.

Book chapters

- K. Sehairi, C. Benbouchama, E.H. Kobzili and F. Chouireb, "Real-Time Implementation of Human Action Recognition System Based on Motion Analysis," in Artificial Intelligence and Computer Vision, ed: Springer, 2017, pp. 143-164.

Conferences

- K.Sehairi, C.Benbouchama, F.Chouireb, "Handel-C Implementation on FPGA of Real Time Motion Detection", Circuits, Systems, Signal Processing, Communications and Computers, Athens, Greece, November 28-30, 2014.

- Kamal Sehairi, Fatima Chouireb, Jean Meunier, " Comparison Study Between Different Automatic Threshold Algorithms for Motion Detection" The 4th International Conference on Electrical Engineering – ICEE'2015 December 13-15th, 2015, Boumerdes, Algeria

- K.Sehairi, C.Benbouchama, F.Chouireb, Kobzili el houari, "FPGA Implementation of Real-Time System for Detection of Human Activity", 3rd International Conference on Automation, Control Engineering and Computer Science, March 20-22, 2016 - Hammamet, Tunisia

ACKNOWLEDGEMENTS

Thanks and praises be to Allah the Almighty who gave me health strength and patience to complete this research study.

I am truly indebted and thankful to my supervisors Pr. Fatima Chouireb and Pr. Jean Meunier whose encouragement, supervision and support from the preliminary to the concluding level enabled me to develop an understanding of my research study. I gratefully acknowledge the funding received towards my PhD internships from Université de Montréal Département d'informatique et de recherche opérationnelle (DIRO). I owe sincere and earnest thankfulness to Dr. Fatiha Benkouider for her support throughout this research study.

I would also like to thank my teachers the Pr. Bouzouad Mouloud, and Pr. Kious Mecheri, Pr. Larbes Cherif and Pr. Djafer Djelloul for honoring me by accepting to be members in the jury of my thesis and help reviewing and improving this work.

It is imperative also to mention the English professors Mr. Mustapha Gasmi, Mme. Tiriri, Mme. Ahmin, Mme. Benmoussa and Ms. Nouioua for their help in improving the English of my research papers and this thesis.

My thanks go also to Pr. Saliha Chettih for her unconditional help in making the workstation under my disposal until completion of my work.

To Pr. Larbes Cherif go my thanks for providing the FPGA board to complete my work started at EMP.

So that those that I have unintentionally forgotten and who participated in any way in the preparation of this work, find here the expression of my gratitude.

DEDICATION

This thesis is dedicated to the memory of my father SEHAIRI Hamouda, my source of motivation and inspiration, may Allah have mercy on him

I miss you a lot

To my mother God bless her,

To my lovely wife and my children

To my dear brothers Cheikh-ali and Nabil

and

My beautiful sisters Zineb and Leila

For them I say sorry this thesis took too much of my time

To all my family specially, Mohammed and Khedidja Rennane, Chahra Hamia, Mamma, Rachid, Karim, Karima Beloufa and Rachida Sehairi

To my best friends Bihiou Abdelali, Ali Korichi, Youcef Benmoulai, Yacine Benyahia, Djamel Silaa, Bachbouch Bendjediaa and Hamza Bendaoudi.

To my colleagues in Idmisas group, Yousfi Mohammed, Benbelghith Ahmed, Lefkaier Ibn Khaldoun, Helifa Bachir, Beramdhan Tayeb, Bader Tayeb, Yousfi Belkacem, Chettih Ali and Benhorma Ahmed

And to all my colleagues in the University of Laghouat, École Normale Supérieure, École Militaire Polytechnique, University of Montreal and Polytechnic of Montreal

To all my teachers from the primary school to the university level

To all, I say Thank you.

“Science without religion is lame; religion without science is blind.”
— *Albert Einstein*

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS ET ACRONYMS

3FD	Three frame difference.
ADL	Activities of daily living.
ADVISOR	Annotated Digital Video for Intelligent Surveillance and Optimised Retrieval.
API	Application Programming Interface.
ASIC	Application Specific Integrated Circuit.
ASTNA	Adaptive spatio-temporal neighborhood analysis.
BE	Behavior entropy.
BGS	Background subtraction.
BoW	Bag of words.
BPNN	Back-propagation neural network.
BSIA	British Security Industry Association.
CART	Classification and Regression Trees.
CC	Connected component.
CCD	Charge Coupled Device.
CHMM	Coupled Hidden Markov Model.
CLB	Configurable Logic Bloc.
CNIL	Commission nationale de l'informatique et des libertés.
CNN	Convolutional neural network.
CORDIC	COordinate Rotation DIgital Computer.
CPLD	Complex Programmable Logic Device.
CPU	Central processing unit.
CVBS	Chroma Video Blanking Synchro (Composite video).
DAC	Digital Analogue Converter.
DCM	Digital Clock Manager.
DK	Design Kit.
DMA	Direct Access Memory.
DP-GMM	Dirichlet process GMM.
D-RGB	Depth Red Green Blue camera.
DSP	Digital Signal Processor.
DSTEI	Difference-based spatio-temporal entropy image
DTW	Dynamic Time Warping.
EDIF	Electronic Design Interchange Format.
EEG	Electroencephalography-based systems.
Eig-BG	Eigen-Backgrounds.
EM	Expectation maximization.
ESL	Electronic System Level.
F1	F1-score (F-measure).
F2F	Frame to Frame Tracker.
FC-CNN	Fully connected convolution neural network.
FD	Frame difference.
FF	Flip Flops.
FIFO	First Input First Output.
FMTG	Forgetting morphological temporal gradient.
FN	False negatives.
FNR	False negative rate.
FP	False positives.
FPGA	Field Programmable Gate Array.
FPR	False positive rate.

FSM	Finite State Machine.
FTSG	Flux tensor with split Gaussian models.
FTU-2	Flash Transfer Utility.
GMM	Gaussian Mixture Model.
GOPS	Giga operation per second.
GPA	Generalized Procrustes analysis.
GPP	General Purpose Processor.
GPU	Graphical processor unit.
GUI	Graphical User Interface.
H/W	Height/width.
HAR	Human activity recognition.
HDMI	High definition multimedia interface.
HMAX	Hierarchical Model and X.
HMM	Hidden Markov Model.
HoDG	Histogram-oriented depth gradients.
HOFM	Optical flow orientation and magnitude.
HOG	Histogram oriented gradient.
HOOF	Histogram of oriented optical flow.
HSV	Hue, Saturation, Value.
HTML	Hypertext Markup Language.
I/O	Input/Output.
ICDM	International Conference on Data Mining.
ICM	Iterative conditional mode.
IDE	Integrated Development Environment.
IOB	Input Output Bloc.
IP	Intellectual Property.
IR	Infra-Red.
ISE	Integrated Software Environment.
ISODATA	Iterative Self-Organizing Data Analysis Technique.
JC	Jaccard coefficient.
JPEG	Joint Photographic Experts Group.
KDE	Kernel density estimation.
kNN	k-nearest neighbor.
K-SVD	K-singular value decomposition.
LBP	Local Binary Pattern.
LC	Logic cells.
LCD	Liquid Crystal Display.
LSTM	Long short term memory.
LUT	Look Up Table.
MAP	Maximum a posteriori.
MDT	Mixture of dynamic texture.
ME	Maximum Entropy.
MHI	Motion history images.
MLP	Multi-layer perceptron.
MMI	Monolithic Memories.
MMX	MultiMedia eXtended.
MoG	Mixture of Gaussians.
MoG-PSO	Mixture of Gaussians with particle swarm optimization.
MRF	Markov Random Field.
MRFMD	Markov Random Field Motion Detection.
MSI	Medium Scale Integration.

NGC	Native Generic Circuit.
NGO	Native Generic Object.
NN	Neural network.
NTSC	National Television Systems Committee.
OAM	Overall average metric.
OPA	Ordinary Procrustes analysis.
PAL	Platform Abstraction Layer.
PAL	Phase Alternation Line.
PAL	Programmable Array Logic.
PBAS	Pixel-Based Adaptive Segmenter.
PC	Personal Computer.
PCA	Principal component analysis.
PCC	Percentage of correct classification
PCI	Peripheral Component Interconnect.
pdf	Probability density function.
PDK	Platform Developer's Kit.
PETS	Performance Evaluation of Tracking and Surveillance.
PLD	Programmable Logic Device.
Pr	Precision.
psC	Parallel and Synchronous C.
PSL	Platform Support Library.
PSP-MRF	Probabilistic super-pixel Markov random fields.
PTZ	Pan Tilt Zoom camera.
PWC	Percentage of wrong classification.
RAF	Running Average Filter.
RAM	Random Access Memory.
RBF	Radial basis function.
RC	Rank across categories.
Re	Recall.
RGA	Running Gaussian average.
RGB	Red Green Blue.
RMI	Recurrent motion image.
RMoG	Region-based mixture of Gaussians.
ROM	Read Only Memory.
SA-C	Single-Assignment dialect of the C programming language.
SACON	Sample consensus.
SBE	Scene behavior entropy.
SC-SOBS	Spatially coherent self-organized background subtraction.
SECAM	Sequential Color with A Memory.
SFM	Social force model.
SGMM	Splitting Gaussian mixture model.
SGMM-SOD	Splitting over-dominating modes GMM.
SIFT	Scale invariant feature transformation.
Simp-SOBS	Simplified self-organized Background Subtraction.
SL-ICA	Subspace learning independent component analysis.
SL-INMF	Subspace learning incremental non-negative matrix factorization.
SL-IRT	Subspace learning using incremental rank-tensor.
SL-PCA	Subspace learning using principle component learning.
SMO	Sequential Minimal Optimization.
SOBS	Self-organized Background Subtraction.
Sp	Specificity.

SRAM	Static Random Access Memory.
SSD	Sum of squared differences.
SSI	Small Scale Integration.
S-T MRF	Spatio-temporal Markov random field.
S-TAP-MoG	Spatial-Time adaptive per pixel mixture of Gaussian.
STEI	Spatio-temporal entropy image.
STFT	Short-time Fourier transform.
STMM	Student-t mixture model.
SuBSENSE	Self-Balanced SENsitivity SEgmenter.
SURF	Speeded Up Robust Features.
SUV	Sport utility vehicle.
SVD	Singular value decomposition.
SVM	Support vector machine.
TDNN	Time Delay Neural Network.
TFT	Thin Film Transistor (flat screen).
TN	True negatives.
TP	True positives.
USB	Universal Serial Bus.
VGA	Video Graphical Adaptor.
VHDL	Very High-speed-integrated Circuit Hardware Description Language.
ViBE	visual background extractor.
VLSI	Very Large Scale Integration.
W^4	Who? When? Where? What? System.
YC	Yule coefficient.
YC_bC_r	Luminance Chrominance blue Chrominance red.

Introduction

“A picture is worth a thousand words”

Nowadays, the world is increasingly becoming complex in its infrastructure and its interactions, which has the effect of increasing the number of tragic accidental or intentional events. In return, the companies are also more demanding in terms of safety and prevention and exploit technological advances to meet these needs in the best possible way. Thus we are witnessing today the proliferation of digital video surveillance systems designed and installed in residential homes, public areas or business (banks, shopping malls, factories, airports, schools, residential towers, bus stations, railways, highways, etc.). Video surveillance systems are also extensively used in the statistical analysis of attending a place, monitoring of epileptic patients in hospitals, or billing of vehicles at toll highways.

These systems are mostly used in order to assist the security staff in its work. In their basic form, they have functions limited to the capture, transmission, storage and display of visual data at the monitoring center; they may incorporate hundreds of cameras. These systems generate a huge amount of video information that exceeds the security guards monitoring capabilities. To solve this problem, intelligent video surveillance aims to treat the captured video, using sophisticated algorithms, and retain only the relevant data for security.

Intelligent video surveillance is based on systems that automatically incorporate advanced technologies from several areas; sensors design, signal processing, artificial intelligence and data mining. The aim is to extend the spectrum of video surveillance application by allowing to:

- Predict incidents by detecting suspicious behavior and trigger alarms in real time. As examples of prediction we can cite the following situations: a moving intruder, an abandoned bag, a disappeared piece or object, a vehicle exceeding a maximum speed, a badly parked vehicle, a full room or a chain too long in airport or supermarket, fall of a person and detection of any behavior deemed abnormal.
- Assisting and improving investigative operations by conducting content-based searches, spatio-temporal tracking, and automatic video extractions.

In order to achieve these goals, the intelligent video surveillance system must go through a series of steps. Generally all intelligent video surveillance systems run in three steps: motion detection and tracking, object identification and action recognition.

Motion detection consists of extracting the foreground by eliminating all static objects; the result of this step contains the silhouette of the moving objects which makes it very sensitive and very important for all following steps. In general, this step is hard to achieve due to the

different challenges presented by the observed scenes like weather change, dynamic backgrounds or the type of the used sensor. Most of all motion detection methods execute in five steps; background initialization, background modelling, background maintenance (update), foreground segmentation and thresholding.

The next stage is the object identification; the aim of this stage is to classify the moving regions and recognize them, for example: vehicles, pedestrians, animals. In order to achieve that, a set of features has to be extracted which can be upgraded to describe people appearance (using semantic attributes such as gender, clothes, accessories…), identify the subject face or identify the license plate of vehicles.

The last and hardest stage is the action recognition stage, because it involves the recognition over time, which requires other techniques and high computational systems. The aim of this stage is to identify the behavior of the moving object, for example simple actions like change in direction, increase of speed, bend to pick something, or complex actions like person falling, people fighting or jumping over barrier.

Historic of video surveillance systems

Video surveillance was first developed in the United Kingdom in response to the IRA (Irish Republican Army) attacks. The first experiments in the UK in the 1970s and 1980s led to large-scale programs in the early 1990s [1]. This success led the government to make a campaign among the population, and launched a series of camera installations. Today, cameras in the United Kingdom cover most of the city centers, and many stations and car parks. According to British Security Industry Association (BSIA), in 2013, the number of CCTV surveillance public and private cameras was between 4 million and 5.9 million in the UK. Another report states that more than 500.000 cameras are installed in London [2]. Due to its importance and its success especially in public security, other countries like France have started installing video surveillance systems. In 2012 the CNIL (Commission nationale de l'informatique et des libertés) estimated that the number of cameras used for public security was more than 935000 cameras. These cameras are present in various places such as airports, railway stations, roads, public transport.

In Canada, the first open street surveillance camera was installed in 1990, the city of Sherbrooke, installed the first surveillance camera in a public space for the purpose of curbing delinquent behavior [3]. Now, many cities and municipalities are using surveillance open street camera like Montreal, Toronto and London.

In Spain, the government has set up a system that allows the whole Strait of Gibraltar to be observed, in order to deal with illegal immigration. With this tool, the Coast Guard can spot any boat that sails on the Strait, that is, within a radius of approximately 14 kilometers, day and night.

In Algeria, video surveillance systems were installed for the first time in 2003 in the capital city, with more than 25 cameras. In 2015, their number increased to more than 3000 installed in many places known that are overcrowded like “Place de la Concorde civile” (ex-1 May), “Place des Martyrs” and the highway road. In parallel, other large cities like Oran and Constantine start installing open street video surveillance systems[4].

Why installing video surveillance systems?

The reasons for the installation of video surveillance systems are diverse. However, public security and the protection of properties are key elements in the justification of the use of video surveillance.

The implementation of video surveillance makes it possible to improve the management of incidents as well as an increase in the efficiency and the speed of intervention; for example, in the prevention of suicide, falls or even in accidents that could occur on highway.

In addition to the goal of securing people and infrastructures, the new video surveillance technologies now make it possible to count clients in a shop, or vehicles entering and leaving a company, to embed video image the amount of a receipt or the information of a badge or access card, to automatically recognize vehicle license plates, to automatically report a lost, abandoned or stolen object, to inform car drivers in real time about traffic conditions.

But the most measurable effect of surveillance cameras is not to discourage crime, but to better detect it and prosecute the perpetrators. Many cases of crimes have been resolved through the recordings provided by the surveillance cameras. For example, after the London underground attacks of July 7, 2005, surveillance camera recordings were used to identify terrorists.

According to the police (in England), the use of cameras on the roads has enabled a significant reduction in speed excess and accidents. A report shows that there was a 68% reduction in people killed in one of the "test" regions, and in all test regions accidents have

also decreased. The police report shows also that the use of license plate recognition software had drastically reduced the number of offenses and allowed a suspicious car to be tracked.

Purpose of this work and contribution

Our goal in this work is to develop a full automated intelligent video surveillance system using a fixed camera that can recognize abnormal or suspicious actions. The goal is achieved by performing the state of the art of the different steps in video surveillance system, investigating and benchmarking the existing methods, proposing new algorithms, enhancing the recognition rate of our system, and finally trying targeting a hardware system.

That is in this context that our work contributed in the field of video processing, more precisely in the field of video surveillance by:

1. Defining the appropriate threshold method for motion detection (frame difference and background subtraction methods).

2. Contribute in the analysis and benchmarking of different motion detection methods, see CD.net 2014 challenge, section results at http://changedetection.net/.

3. Designing and implementing a real time video surveillance system on FPGA using high level language that can detect behavior change using motion analysis.

4. Developing a system for detecting fall for elder people by proposing a new method for estimating the position of the head.

This thesis is organized as follows:

In chapter 1, we present the state of the art of motion detection, identification of moving objects and recognition of actions, and the different techniques and algorithms used. We will also present the different real-time implementations of these methods, focusing on the implementations that use FPGA and GPU circuits.

In chapter 2, we propose a comparative study between different global thresholding methods applied in various differential motion detection algorithms. The purpose of the comparison is to define the appropriate automatic threshold method for surveillance applications, both in indoor and outdoor areas. In order to achieve that, six threshold methods have been tested on different differential motion detection algorithms, using four scenes with

different complex backgrounds. A pixel-based evaluation method has been done to determine the best combination.

In chapter 3, we perform a study to compare several change detection methods, for a monostatic camera, and identify the best method for different complex environments and backgrounds in indoor and outdoor scenes. To this end, we used the CDnet video dataset as a benchmark that consists of many challenging problems, ranging from basic simple scenes to complex scenes affected by bad weather and dynamic backgrounds. Twelve change detection methods, ranging from simple temporal differencing to more sophisticated methods, were tested and several performance metrics were used to precisely evaluate the results. Because most of the considered methods have not previously been evaluated on this recent large scale dataset, this work compares these methods to fill a lack in the literature, and thus this evaluation complements the previous comparative evaluations. Our experimental results show that there is no perfect method for all challenging cases; each method performs well in certain cases and fails in others. However, this study enables the user to identify the most suitable method for his or her needs.

In chapter 4, we propose a PixelStreams-based FPGA implementation of a real-time system that can detect and recognize human activity by applying a finite state machine. The implementation was done using high level language. The first part of this work details the hardware implementation of a real-time video surveillance system on FPGA, including all the stages, i.e., capture, processing, and display, using DK IDE. The targeted circuit is an XC2V1000 FPGA embedded on Agility's RC200E board. In the second part of this work, we propose a GUI programmed using Visual C++ to facilitate the implementation for novice users. Using this GUI, the user can program/erase the FPGA or change the parameters of different algorithms and filters. The PixelStreams-based implementation was successfully realized and validated for real-time motion detection and recognition.

In chapter 5, we present an automatic intelligent video-based fall detection system. First, we extract the silhouette of the person using a background subtraction technique, then a set of features is measured to define if a fall happened, for that a new technique is presented to estimate the head position, this algorithm is tested on the L2ei dataset where more than 2700 frames have been labelled in order to train three different machine learning algorithms. The results shows that our system can predict the correct class with an accuracy that can reach 99,61% and with a maximum global error of 1,5%.

References

[1] V.Gouaillier,A.Fleurant, «La vidéosurveillance intelligente : promesses et défis», Technopôle Défense et Sécurité, Québec, 2009.

[2] M.McCahill, C.Norris. « CCTV in London».5th Framework Programme of the European Commission. Hull, United Kingdom, June 2002.

[3] S. C. A. Network and W. Deisman, A Report on Camera Surveillance in Canada: Part One: SCAN, 2009.

[4] M. Menhaouara, «LA vidéosurveillance urbaine en Algérie. A quel coût et pour quelle efficacité ? », 14/12/2015, le soir d'Algérie http://www.lesoirdalgerie.com/articles/2015/12/14/article.php?sid=188525&cid=41, 13/10/2017.

Chapter I

Video surveillance systems

State of the art

The one who follows the crowd will usually go no further than the crowd; the one who walks alone is likely to find herself in places no one has ever been before.

Albert Einstein

1. INTRODUCTION

Intelligent video surveillance systems incorporate many fields of sciences, starting by physics and optics for sensors, mathematics for data mining and matrix manipulation, informatics for coding and code optimization and electronics for hardware implementation and verification. This made intelligent video surveillance systems hard to achieve and need a lot of knowledge in many areas. In general, these systems execute in three stages: motion detection and segmentation, identifying the moving objects, recognizing action and understanding the behavior of the object. The first stage consists of detecting the existence of motion in the video stream while tracking the detected object. It is highly dependent on the application, for which a video surveillance system is used, but most of these systems perform this task on gray scale in order to reduce the amount of data to be treated, nevertheless this operation can decrease the quality of such systems. Other systems use different color spaces like RGB, HSV or another type of sensors like Depth camera (D-RGB, Kinect) or thermal cameras. In our study, we will investigate different detection and segmentation methods in order to elaborate a benchmark that can help the user to choose the best method in terms of result, its adaptation to the change of the environment, the consumption of the hardware resources and the complexity of calculation. It should be noted that this step is the most important in video surveillance systems, because the wrong segmentation of the moving object will alterate all subsequent steps.

The second stage focus on the identification of objects, i.e. classification of moving objects and description of their appearance, for example if we want to analyze the behavior of a human being, we will be interested in movement of persons neglecting the movement of animals or cars; if we are interested on that man with a red t-shirt and black backpack we would be concerned on identifying that person using semantic attributes like gender, clothes, accessories,… In this context several methods has been proposed such as shape-based methods, texture-based methods, gait and motion-based methods or semantic-based methods.

The last stage is action recognition and behavior understanding, which constitutes the high level of video surveillance (Fig.I.1). This stage requires a lot of information in order to train the system to recognize correctly the actions. The recognition is achieved using feature extraction, motion analysis and machine learning methods. Fig.I.2 represents the flowchart of a standard intelligent video surveillance system. It should be noted that many motion detection

methods use learning techniques to segment moving objects (see. Chapter III), further, in case there is a network of cameras, a data fusion step is required for target tracking purpose.

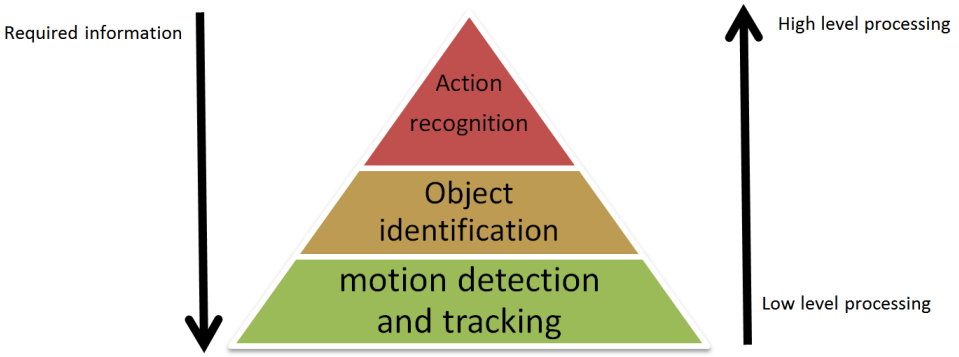


Fig.I.1. Stages of an intelligent video surveillance

To carry out our work, it was essential to review the different methods of these stages, and since our work focus on realization, we have also detailed the different circuits used in implementation of such systems.

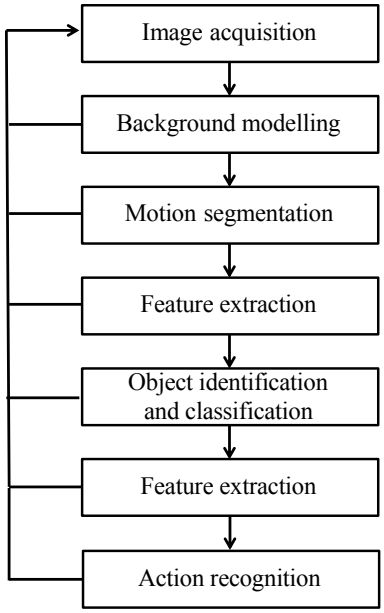


Fig.I.2. Flowchart of a standard intelligent video surveillance system

In this chapter, we start by presenting the different circuits dedicated to real-time implementation, the difference between them and the criteria for selecting one of these circuits. Then, we review the different methods of motion detection and segmentation; we discuss the resource consumption of each method, and cite some work done on hardware circuits. We also define ‘suspicious behavior’ and the different criteria set up to detect a

change in behavior, we cite the different methods used, and we present some work and projects carried out for the detection of behavioral change.

2. HARDWARE CIRCUITS DEDICATED TO REAL TIME IMPLEMENTATION

To ensure proper functioning of a video surveillance system, it must operate in real time, which consists of having the same image stream between the video source (input) and the monitor (output), this means that the processing time should not exceed the time of image acquisition, which is often 25 frames per second or 40 ms (Fig.I.3).

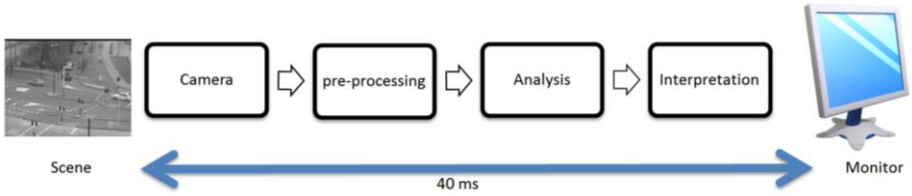


Fig.I.3. The minimum required time for real time video surveillance

However, the great mass of data that carries a video signal and large processing chains for these video surveillance systems often exceeds the processing capacity of a conventional machine GPP (General purpose processor). For that reason, we require specific circuits that can accelerate processing such as Application Specific Integrated Circuits (ASICs), Digital Signal Processors (DSPs), Graphical Processor Units (GPUs), Field Programmable Gate Arrays (FPGAs), parallel processors and heterogeneous processors… However video processing developers often have to compromise between flexibility and performance due to the limited available resources in the hardware. If we have to obtain a good accuracy system we have to increase the model size and data, which will take more hardware resources (RAM, input output and power), however, increasing the latter will reduce the speed, Fig.I.4.

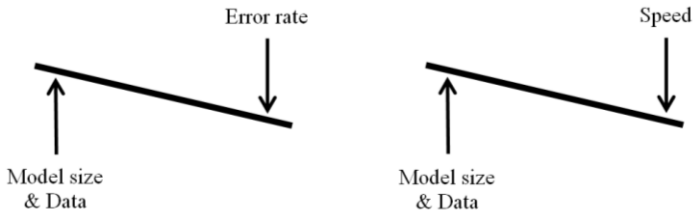


Fig.I.4. Model size/Data and speed dilemma

ASICs have long been the most suitable technology for performing applications requiring high performance, in a small size, in terms of computing power compared to power consumption, weight and volume. The performances of the ASICs are much better than those of traditional microprocessors and microcontrollers. The ASICs, however, suffer from some

disadvantages: the design costs and the initial investment in production are considerable. For an ASIC component to have a tolerable price, it must be mass-produced, so that it can amortize the initial investment. Also, ASICs require a lot of time in development, testing and production and cannot be reconfigurable. As a result, ASIC technology is only used in specific projects.

DSPs are inexpensive, flexible and can be used in many applications. However, these processors can deliver only limited computing power in real time because of their limit in parallel processing. To enhance the performance of these systems most DSP manufacturers introduce multicore and heterogeneous systems like the Davinci or keystone multicore Texas instruments system.

GPUs are becoming recently widely used because of their performance unlike other circuits. These circuits are designed for complex mathematical and geometric calculations that are necessary for graphics rendering. These systems execute tasks in full parallelism; they could contain hundreds of cores in a single chip. For example the Nvidia Volta has more than 5000 cores with more than 40 billion transistors.

Despite the great performance of these processors, these beasts are power hungry which make them unsuitable for embedded systems, additionally, they require an efficient cooling system.

FPGA is an integrated circuit consisting of a large number of logic elements interconnected by programmable routing matrix. This structure allows the FPGA to emulate any circuit, on condition that it is not too big and do not use up the logic and routing resources of the FPGA. Xilinx and Intel-Altera are the worldwide leaders of manufacturing of these circuits. They propose different architectures and families. In general all FPGA contains these elements: multipliers (DSP multipliers in recent FPGAs), block RAMs, digital clock manager (DCM), Input Output blocks (I/O B) and configurable logic blocks (CLB) which consist of logic cells (LC), logic gates, registers and multiplexors.

This structure allows the FPGA to execute in full parallelism all kind of tasks and unlike the ASICs, these circuits are reprogrammable, they can be erased and written several times; additionally, their performances compared to power consumption, volume and weight are increasingly approaching those of ASICs, which make them well adapted to video processing applications and embedded systems.

Table I.1 Comparison between different hardware circuits

Characteristics	ASICs	GPPs	DSPs	GPUs	FPGAs
Performance	Very high	Moderate	Moderate	Very high	High
Size and weight	Small	Large	Moderate	Moderate	Moderate
Energy consumption	Low	High	moderate	very high	moderate
Integration	System on chip	Other component are required	System on chip	Other component are required	System on chip
Programmability	Non programmable	Programmable	Programmable	programmable	programmable
Required expertise for implementation	High	Moderate	Moderate	Moderate	Moderate
Initial investment (according to the user)	Very high	Low	Low	Moderate	Low

To implement a video processing application, we must first choose an adequate hardware platform according to several criteria such as price vs. performance, portability, algorithm complexity, chosen architecture, required memory bandwidth, time of development and finally real time constraints. Table I.1 presents a simple comparison between the different discussed circuits.

3. MOTION DETECTION

Detection of moving objects in a scene is a necessary task in several areas such as video surveillance (intrusion, road traffic or target pursuits), robotic tasks (obstacle detection, missile guidance), medical imaging .... Thus, a large number of motion detection algorithms have already been proposed. To avoid redundancy, section 2 in chapter III presents in detail the state of the art and the different categories of motion detection methods; however we will focus here on real time implementation of these methods on different hardware circuits.

Menezes *et al.* [1] implemented on FPGA a simple background subtraction method to detect vehicles in roads; the aim was to compare a software implementation on PC with Intel Celoron processor with 2.3 Ghz and a hardware implementation on Cyclone II FPGA. The results showed that the hardware implementation is 7.5 times faster than the software implementation.

In another work, Gorgoń et al. [2] used the Handel-C language and the PixelStreams library of the DK design suite to model the background image by calculating the average of each pixel of the previous images, and then using the Sum of the Absolute value of Differences

(SAD), Fig.I.5. The implementation was done on the Agility RC300E board equipped with VirtexII-V6000 FPGA, the number of consumed CLB was 11%, 5% in block RAMs, and 32% of the I/O blocks.

In recent work, Kryjak et al. [3] presented an FPGA-based system that can generate the background and extract the foreground, the system can process HD color video stream in real time with 60 frames per sec. The project was synthesized for a Virtex 6 FPGA fitted in ML605 board using Xilinx ISE 13.4 Design Suite. The resource usage was 16% of CLBs, 4% of DSP multipliers and 9% of BlockRams.

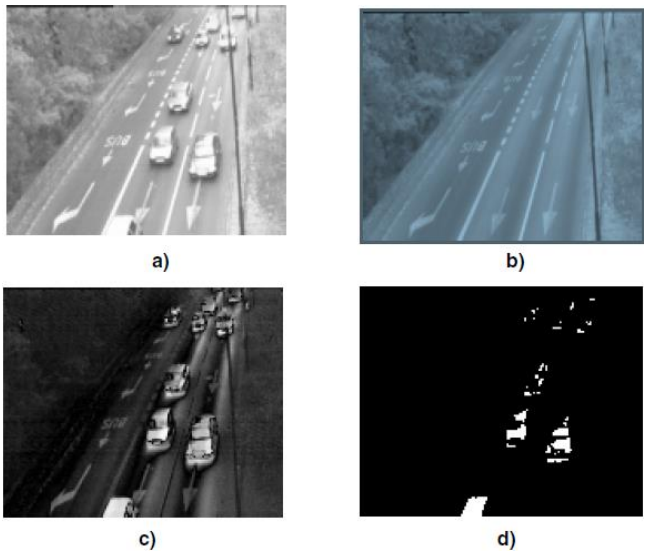


Fig.I.5. Detection result: a) individual video frame; b) generated background; c) differential image between actual frame and background; d) the result of segmentation by thresholding [2]

In another work, Liu et al. [4] proposed a joint optical flow and principal component analysis (PCA) approach to detect motion from airborne videos, Fig.I.6. The algorithm was implemented on the GPU NVIDIA GeForce GTX480 with 1.5 GB and compared to CPU implementation on Intel core i7-860 2.80 GHz and 4 GB memory. The results showed that the processing on GPU can be 10 times faster than the CPU for PCA and was reduced by more than half for the optical flow algorithm.

(a) (b)

Fig.I.6. Implementation result of the joint method applied on airborne image: (a) the input frame, the moving vehicles are highlighted with yellow, (b) detection result, the three vehicles are well detected (in red) but there were false alarm pixels [4]

Beleznai *et al.* [5] presented a GPU-accelerated method for motion and pedestrian detection. The motion detection algorithm was based on the codebook algorithm and the implementation was done on the NVIDIA GeForce 9800GT GPU using CUDA, Fig.I.7. The results showed that the implementation can run on real time with a rate of 61 fps for a video resolution of 720x576.

Fig.I.7. Detection results on the Caviar dataset [5]

Many other hardware implementations can be found in [6-10].

4. OBJECT CLASSIFICATION AND IDENTIFICATION

After the detection of movements in sequences of images or video stream, the object identification and behavior understanding follow naturally. This stage represents a high-level task in video surveillance applications. The aim of object identification is to learn and segment automatically all the moving objects in the scene and classify them into different groups (human beings, vehicles, animals,…). This task can be further detailed to recognize different types in the same group, for example, different types of vehicles (vans, SUVs, cars and buses,…), see Fig.I.8.

Hu *et al.* [11] categorized object identification and classification methods in two groups; shape-based methods and motion-based methods. Another recent survey done by Paul *et al.* [12] add to these categories, the texture-based methods. To these categories we can add the 3D modeling [13-15] and semantic features (gender, race, hair color, accessories,…) [16-18] which have been used also for object classification. However these techniques require a high image quality and multi-view camera system for best classification.

In the shape-based methods we are interested in extracting different shape features from silhouettes of the moving objects. For example: Lin and Wei [19] used the width/height ratio to classify moving objects into three classes: pedestrian, vehicles, motorcycles and others.

Lipton *et al.* [20] showed that the pedestrian with its complex shape has a large dispersedness compared to vehicles; this metric has been used to classify moving regions into three categories: human, vehicles or clutter.

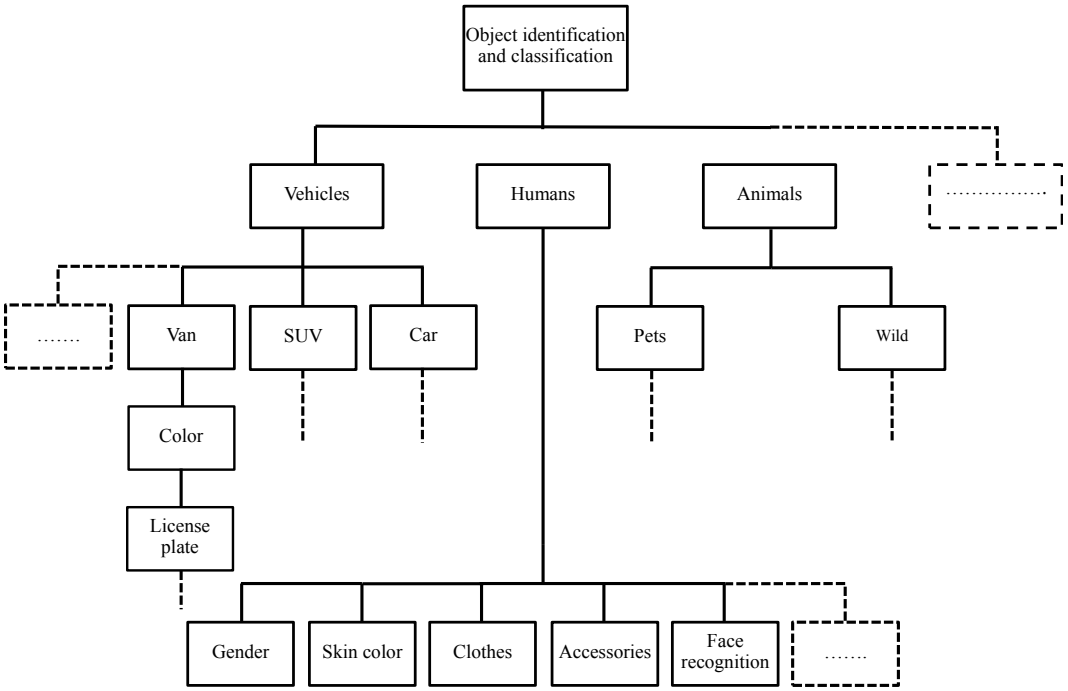


Fig.I.8. Example of object identification and classification applications

Collins *et al.* [21] combined the last two features with the image blob area to develop a visual surveillance system that can differentiate person, crowd, vehicle or clutter. Different shape features can be used for object classification and identification like the perimeter, roundness, convexity, compactness [22,23], the ellipsoidal region, flattening, linear eccentricity, numerical eccentricity and bending energy[24-26].

In motion-based techniques, we can identify and classify objects from their movements; in general, we are interested in periodic movement of the non-rigid objects or their parts, for example Cutler and Davis [27] used short-time Fourier transform (STFT) with autocorrelation for periodicity detection. The results showed that the algorithm can classify human and animals. In another work, Javed and Shah[28] developed the recurrent motion image RMI to calculate repeated motion of objects. The result was similar to motion history images (MHI) or energy motion images developed in [29]; the tests were carried on the PETS2001 dataset in order to recognize human and vehicles. Ran *et al.* in [30] examine the periodic motion of gait to recognize pedestrians and track them.

Texture-based method uses different features to identify objects, for example Papageorgiou *et al.*[31,32] used the Haar wavelets as features to identify different classes: faces, pedestrian and cars, Apaten *et al.* [33] used different variant of wavelets (Haar, Daubechies, Coiflet, Symlet, Biorthogonal, Fractional causal or generalized) to classify 5 different categories from the Robin competition dataset[1], Belongie *et al.*[34] developed a new approach for measuring shape similarity. They used the histogram of a log polar plot of the shape boundary to generate the shape context for that particular boundary point (Fig.I.9). Corresponding points on two similar shapes are supposed to have the same shape context that facilitates shape recognition [35]. Wang *et al.*[36] used this algorithm to classify moving object into two classes vehicle and pedestrian.

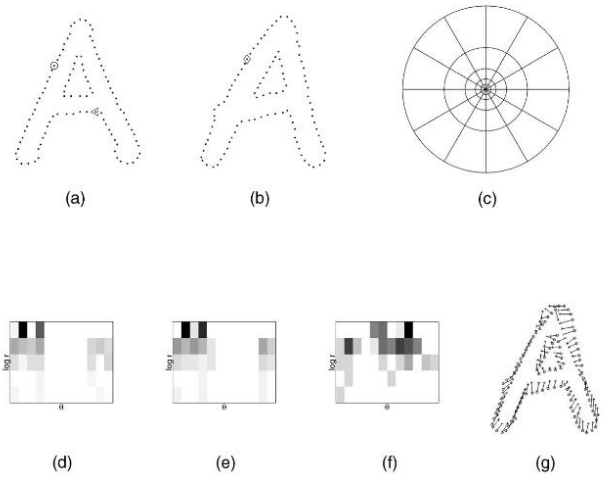


Fig.I.9. Shape context computation and matching: a) and b) Sampled edge points of two shapes. c) Diagram of log-polar histogram bins used in computing the shape contexts, d),e),and f) Example shape contexts for reference samples marked by ○,◊, ◁ , g) [34]

Similar to shape context, Lowe [37] presented another descriptor that performs reliable matching between different views of an object by transforming image data into scale-invariant coordinates relative to local features, he called it the Scale Invariant Feature Transformation descriptor (SIFT). Mikolajczyk *et al.*[38] join this feature with other features to detect human in single images.

Based on SIFT, Dalal and Triggs [39] proposed the Histogram of Oriented Gradient (HOG). This descriptor operates by dividing an image region into a number of cells. For each cell a 1D histogram of gradient directions over the pixels in the cell is calculated [40]. Dalal and Triggs used this descriptor to detect human in images, Negri *et al.*[41] compared this

[1] http://robin.inrialpes.fr/datasets.php

descriptor with the Haar wavelet in the aim of designing a robust system for vehicle classification.

Local Binary Pattern (LBP) is the most popular feature, described by Ojala *et al.*[42]. It is a modified version of the Wang and He [43] descriptor. First, it compares the pixel center with its neighbors. The result is thresholded (Fig.I.10.b) then multiplied with weights of order 2^d (where d represents the order of the pixel neighbors, Fig.I.10. c). Finally, the neighbors are summed to obtain LBP of that pixel (Fig.I.10.d).

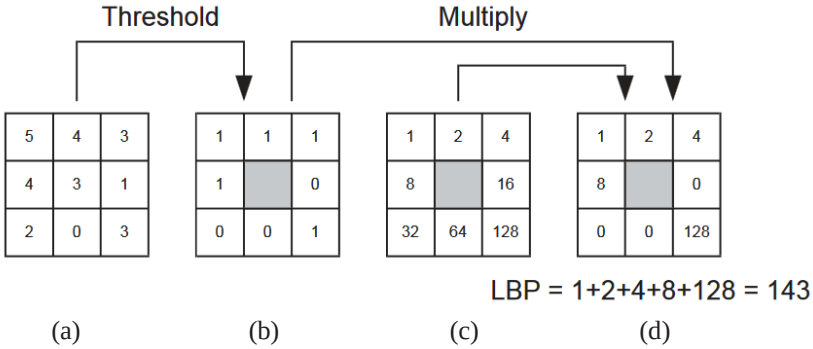


Fig.I.10. Calculating the LBP texture [44].

In their paper, Ojala *et al.* [42] compared the LBP feature with different other texture measures for object classification (Gray-level difference method, Law's texture method, Center-symmetric covariance measure and complementary feature pairs). A classification example for visual surveillance application can be found in [45,46] and in which they use different variants of the LBP.

Other methods that use self-similarity descriptors to identify objects in scenes can be found in [47-49]. We should note that many of these features [50,51] can be also used for background modelling and foreground detection since the motion detection task can be considered as a classification problem.

After feature extraction, the step of classification begins; in its simple way, we can set a threshold to determine the class, but it is not always the case especially with the texture-based approach or when we have high feature dimension and multi-target classification. For that, many data mining techniques and machine learning methods were developed to solve the discrimination problem. In [52] Wu *et al.* presented the top 10 data mining algorithms identified by the IEEE International Conference on Data Mining (ICDM), the C4.5 (decision tree) [53], K-means [54,55], support vector machines (SVM) [56], the A priori algorithm [57], Expectation maximization algorithm (EM) [58,59], PageRank [60], AdaBoost [61], k-nearest

neighbor kNN [62,63], Naïve Bayes [64,65] and Classification and Regression Trees CART[66]. For each algorithm, they gave a description, its limitation and its different versions. However, different new techniques and new improvements for the existing ones have been done since this last report. We can state as examples the linear regression[67-69], the logistic regression [70-72], Kernel principle component analysis PCA[73-76], K-singular value decomposition K-SVD [77,78], neural networks[2] (Back-propagation neural network BPNN [79,80,81], recurrent neural networks [82], deep convolutional neural networks [69,81,83-87], long short term memory recurrent network LSTM-CNN [88-92], deep belief networks[93], AlexNet[94], VGG[95] and GoogLeNet[96]). It should be noted that unlike the hand-crafted (hand-designed) feature generation in traditional object and action recognition systems, the CNN-based systems can generate automatically the features. Furthermore, many libraries for different CNN architectures exist like Caffe[3] and Caffe2[4], Theano[5], Tensorflow[6]. A pre-trained version of these frameworks with ImageNet[7] dataset also exists (to overcome the time of training). Other data mining algorithms and benchmarks can be found in [97,98].

Many applications and libraries have been implemented for these techniques, like the SVMlibrary[8][99], the WEKA[9] [100,101] and Accord.Net[10] [102].

For implementations on hardware circuits, we can state the work of Meus *et al.* [103] in which they implemented a real time system for pedestrian detection on Zynq FPGA board (ZC 702 board). They used the histogram of gradient with the support vector machine. The authors reported that the computing performance of this implementation was 7.9 GOPS which allows a real time processing for 1280x720@60Hz video stream. The resource consumption was 48% for LUTs, 32.33 for FFs, 24.64% BRAMs and 49% of DSP multipliers.

Kyrkou and Theocharides [104] implemented a visual system for moving object detection and classification. A visual-feature-directed search cascade composed of motion detection, depth computation, and edge detection has been used for detecting and extracting features. A non-linear SVM was used for classifying objects. The implementation was conducted on

[2] http://www.asimovinstitute.org/neural-network-zoo/
[3] http://caffe.berkeleyvision.org/
[4] https://caffe2.ai/
[5] http://deeplearning.net/software/theano/
[6] https://www.tensorflow.org/
[7] http://image-net.org/about-overview
[8] https://www.csie.ntu.edu.tw/~cjlin/libsvm/
[9] https://www.cs.waikato.ac.nz/ml/weka/index.html
[10] http://accord-framework.net/index.html

AVNET industrial video processing board equipped with a Xilinx Spartan 6 LX150T FPGA with interfaces to multiple cameras and high definition multimedia interface (HDMI) sources. The system can process up to 50 1024×768 pixels images per second, with 79% of LUTs, 25% of FF, 83% of Block RAMs and 4% of DSPs consumed.

Hiromoto et al. [105] designed a similar system in which they used the Haar wavelet with the Adaboost classifier. They implemented their system on Virtex 5 (XC5VLX330-2) FPGA. Their algorithm required 30% of available resources on the target FPGA and can processed images with 640×480 at 30fps.

Kryjak et al. [106] implemented a real time system that detects the head and the shoulder of the pedestrian. For that, they used the LBP with an SVM classifier. The hardware circuit used was the Virtex 6 fitted on the Xilinx ML605 board, Fig I.11. The system processed a video stream resolution of 640x480 pixels at 60fps. More implementations can be found in [107-110].

Fig.I.11. The hardware system based on Xilinx ML605 board [106]

Heymann *et al.* [111] presented an optimized technique for generation of the Scale Invariant Feature Transform (SIFT)[37] in order to implement it on hardware circuits. They compared a CPU implementation on Intel Xeon with 3.2GB and GPU implementation on NVIDIA QuadroFX 3400. The results showed that the GPU implementation is 7 times faster than CPU implementation.

Prisacariu and Reid [112] achieved a real time implementation of the HOG algorithm using the NVIDIA GeForce GTX 285 GPU. Testing their implementation on the INRIA person

dataset[11], they achieve a 67x speedup in colour mode and a 95x speedup in grayscale mode, over the CPU implementation.

Table.I.2 presents the different features and classifiers used in the papers stated above.

Table I.2 Examples of used features and classifiers for object classification

Paper	Feature	Classifier
Lin and Wei [19]	H/W ratio	Threshold
Lipton *et al.* [20]	despersedness	Threshold
Collins *et al.* [21]	Despersedness, H/W ratio, Area	Neural network
Javed and Shah[28]	recurrent motion image	Threshold
Papageorgiou et al.[31]	Haar wavelet	SVM
Apaten et al. [33]	Haar, Daubechies, Coiflet, Symlet, Biorthogonal, Fractional causal and generalized wavelet	kNN
Belongie et al.[34]	Shape contexts	kNN
Mikolajczyk et al.[38]	SIFT	Cascade of classifiers, AdaBoost
Dalal and Triggs in [39]	HOG	SVM
Negri et al.[41]	Haar vs. HOG	AdaBoost
Ojala et al.[42]	LBP vs other features	KNN
Oh and Kang [69]	Automatic	CNN
Kim et al. [71]	Haar, HOG, B HOG	Logistic Regression
Khellal et al. [72]	Automatic	CNN
Liang et al. [75]	Haar	KPCA with Fisher discriminant Analysis
Qi et al.[77]	HOG	K-SVD
Zheng et al. [78]	HOG +LBP	K-SVD
Gouiaa and Meunier [81]	Automatic	CNN
Meus et al. [103]	HOG	SVM
Kyrkou and Theocharides[104]	motion detection, depth computation, and edge detection	SVM
Hiromoto et al. [105]	Haar wavelet	AdaBoost
Kryjak et al. [106]	LBP	SVM

It should be noted that many datasets have been developed for benchmarking these different techniques and to provide different challenging situation like occlusion, dynamic background, different sensors (thermal, RGB, Kinect…), and different object size.

Table I.3 shows the most used and known datasets for pedestrian and object detection and identification. Other specialized datasets for face detection and vehicle classification are not covered in this table.

[11] http://pascal.inrialpes.fr/data/human/

Table I.3 Most used datasets for pedestrian detection and identification

Dataset	Description
USC[12] [113,114]	Three dataset A,B and C collected from the internet and the caviar dataset, to evaluate frontal/rear view walking/standing human detection algorithms with and without inter-human occlusion A: 205 images, with 313 humans, B: 54 images, with 271 humans, C: 100 images, with 232 humans
Data61[13] [115]	Contains 25551 unique pedestrians, with over 50000 images
GM-ATCI[14] [116,117]	The dataset was collected using a vehicle-mounted standard automotive 180° fish display camera for evaluating rear-view pedestrian. It contains 250 clips with a total duration of 76 minutes and over 200K annotated pedestrian bounding boxes The test set consists of 6 sets of clips
CVC[15] [118]	The dataset is composed of day and night sets acquired using visible and far-infrared (FIR) camera Training sets: 3695 images in day time and 3390 images during night with around 1500 mandatory pedestrian annotated for each sequence Testing set: around 700 images for both sequences with around 2000 pedestrian during day, and around 1500 pedestrian during night.
Daimler[16] [119]	8.5Gb captured from a vehicle during a 27 min drive through urban traffic, at VGA resolution. The training set contains 15.560 pedestrian samples and 6744 additional full images not containing pedestrians for extracting negative samples. The test set contains an independent sequence with more than 21.790 images with 56.492 pedestrian labels (fully visible or partially occluded)
INRIA Person dataset [39]	A large set of marked up images of standing or walking people collected from several sources, GRAZ.01 dataset, google images, personal recorded images
MIT People Dataset[17] [31]	Raw images scaled to 128x64 for frontal and rear views of pedestrians training set:1527; testing set:600
MIT StreetScenes[18] [120]	A set of images image taken from a Sony DSC-F717 camera around Boston with resolution of 960x1280, the dataset contains 3547 labeled images with nine categories: cars, pedestrians, bicycles, buildings, trees, skies, roads, sidewalks, and stores
CALTECH[19] [121]	Pedestrian detection dataset consists of approximately 10 hours of 640x480 30Hz video taken from a vehicle driving through regular traffic in an urban environment. training set:192000; testing set:155000
ETH dataset[20] [122]	Urban dataset captured from a stereo AVT Marlins F033C camera mounted on a chariot respectively a car, with a resolution of 640 x 480, and a framerate of 13--14 FPS
TUD-Brussels[21] [123]	Dataset with image pairs recorded in a crowded urban setting with an onboard camera. 508 image pairs with 640×480 pixel resolution with 1326 annotated pedestrian
ImageNet	An image database used for the ImageNet challenge with a total number of images 14,197,122 and more that 1000 sunsets
CIFAR 10[22]	The CIFAR-10 dataset consists of 60000 32x32 colour images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images.
CIFAR 101[23]	Pictures of objects belonging to 101 categories. About 40 to 800 images per category. Most categories have about 50 images. he size of each image is roughly 300 x 200 pixels

[12] http://iris.usc.edu/Vision-Users/OldUsers/bowu/DatasetWebpage/dataset.html
[13] https://research.csiro.au/data61/automap-datasets-and-code/
[14] https://sites.google.com/site/rearviewpeds1/
[15] http://adas.cvc.uab.es/elektra/enigma-portfolio/cvc-14-visible-fir-day-night-pedestrian-sequence-dataset/
[16] http://www.gavrila.net/Research/Pedestrian_Detection/Daimler_Pedestrian_Benchmark_D/Daimler_Mono_Ped__Detection_Be/daimler_mono_ped__detection_be.html
[17] http://cbcl.mit.edu/projects/cbcl/software-datasets/PeopleData1Readme.html
[18] http://cbcl.mit.edu/software-datasets/streetscenes/
[19] http://www.vision.caltech.edu/Image_Datasets/CaltechPedestrians/index.html
[20] https://data.vision.ee.ethz.ch/cvl/aess/dataset/
[21] https://www.mpi-inf.mpg.de/departments/computer-vision-and-multimodal-computing/research/people-detection-pose-estimation-and-tracking/multi-cue-onboard-pedestrian-detection/
[22] https://www.cs.toronto.edu/~kriz/cifar.html
[23] http://www.vision.caltech.edu/Image_Datasets/Caltech101/

5. ACTION RECOGNITION AND BEHAVIOR UNDERSTANDING

Understanding human behavior involves analyzing and recognizing the movements of human, and producing a high-level description of actions and interactions. For example: a person entering a subway channel, people fighting or rioting, a person falling, a car going in the opposite direction or exciding the speed limit… It is possible to set criteria to recognize actions that may be considered suspicious or abnormal. These criteria can be more or less complex depending on the observed scene and the previously defined suspicious behavior. For that, we can classify these systems into two main categories:

- Crowd behavior recognition: the role of such system is the surveillance in the first place, but their role can be extended to crowd management and public space design, or to entertainment [124]. For example, monitoring the movement of supporters or fans leaving a stadium or a concert in order to enhance the entrance and the exit gates, the surveillance of participants on strike, the management of Muslims in pilgrimage, the analysis of crowd movement in order to simulate such behaviors in games or movies…

- Disjoint person/object action recognition.

The performance achieved by crowd monitoring systems is currently modest. This issue is still in the research stage and is also the subject of several projects involving large groups and laboratories. Among these projects, there is the European project ADVISOR (Annotated Digital Video for Intelligent Surveillance and Optimized Retrieval) [125,126] for the surveillance of metros. The system is already installed in Brussels and Barcelona. This system offers assistance to operators in charge of safety in the metro. The principle of this system is based on the recognition, reporting and automatic archiving of suspicious and remarkable behaviors. ADVISOR uses the video sources provided by a network of cameras; the physical units of this system are designed by THALES and integrate various software modules. The first element manages the synchronization of video sources. These video sources are digitalized and compressed in JPEG format. Then, these data are processed by a second module provided by KINGSTON University for the purpose of analyzing crowd behaviors. The third module is the motion detection module provided by INRIA. This module detects the movement of people and automatically does the tracking. An action recognition module also provided by INRIA makes a decision on the behavior perceived and can trigger alarms to the controller. The final unit is an archive server provided by BULL. Its function is to save digital

images and videos and at the same time record annotations generated by previous processes. A human-machine interface is designed by Vigitec in order to simplify the use of this system.

Recently, Arikuma and Mochizuki [127] developed a real time crowd detection system in the central research laboratories at NEC Corporation. The system estimates the crowd density level and people number in the goal of estimating the waiting time in queues and to avoid overcrowding. The system can integrate also various types of multimedia analysis engines (MAG1C[24], Fig I.12), such as both video analysis and acoustic analysis, to detect a variety of unusual activities, e.g. a person of interest detection (based on face detection), license plate recognition, unusual crowd behavior detection and emergency, talk sound detection. Arikuma *et al.* [128,129] propose a system that can reduce the network usage between local surveillance hubs by changing bitrates depending on contents of the video feeds. The system was designed to support various camera configurations available in a real surveillance setup by using a middleware called ASCOT (Analysis Control).

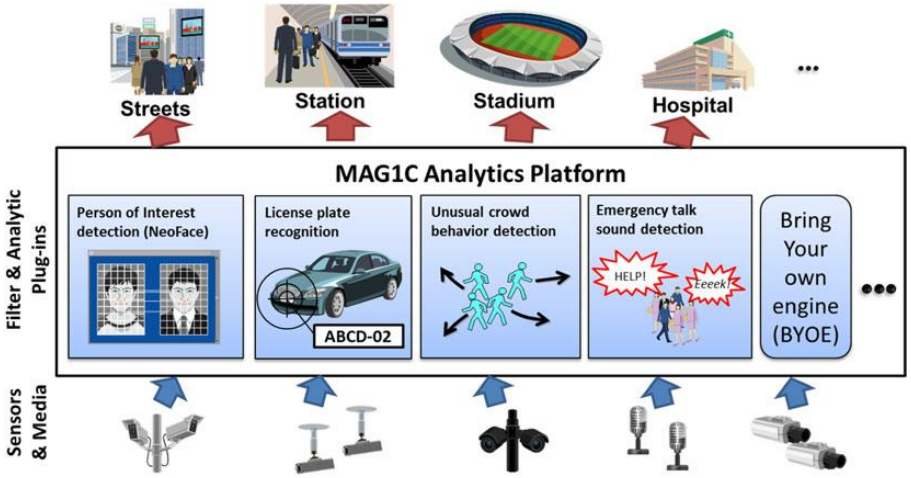


Fig.I.12. Concept of MAG1C Analytics Platform [127]

A research team in Evitech[25] [130] proposed a system for crowd monitoring and management, the system can predict the density of the crowd, its speed and its direction. Other situations can be detected using this system for example the detection of several people who fall, detection of a car/truck or a big object entering the crowd (possibly quickly, endangering people in the crowd), detection of one or several people walking in non-authorized direction, detection of a smoke growing from the crowd or reaching the crowd (fire/choking danger), many people suddenly running, or stopping (fear)…In [130] they discuss how to achieve such systems based on video analytics in six points: 1) conditions of crowd observation, 2)

[24] http://www.nec.com/en/global/solutions/safety/Inter-Agency/index.html
[25] http://www.evitech.com/index.php/en/solutions/crowd-monitoring

subtracting background, 3) putting a cross on each “head”, 4) learning crowd situation patches, 5) analyzing motion, 6) estimating people density.

Li et al. [124] classify the computational modeling of crowd behavior into two major approaches: continuum-based approach and agent based approach. In continuum-based approach, the crowd is treated as a physical fluid with particles, thus a lot of analytical methods from statistical mechanics and thermodynamics are introduced, for example Moore et al. [131] analyze very dense crowd scenes using a hydrodynamics based techniques, considering the movement of people in high-dense crowds as particles in liquids or gases.

Ren et al. [132] introduced information theory and energetics concept to detect abnormal crowd behavior using entropy models. A Scene Behavior Entropy (SBE) is used to denote the dispersion on different directions of crowds and pixels' behavior entropy (BE) distribution to denote interaction information among individuals.

Ihaddadene and Djeraba [133] proposed the motion heat map which is a 2D histogram indicating the main regions of motion activity. This histogram is built from the accumulation of binary blobs of moving objects, which were extracted following the background subtraction method.

In the agent based approach, the movement of each individual pedestrian is considered as autonomous agent who actively sense the environment and make decisions according to some predefined rules. Helbing and Molnár [134] proposed the social force model (SFM). The assumption in this model is that the interaction force between pedestrians is a significant feature for analyzing crowd behaviors. The model proved to be capable of reproducing specific crowd phenomena [124], and many generalization of this feature can be found in [135-136]. More physics-based methods for group and crowd analysis can be found in this review [137].

Optical flow combined with clustering based methods has also been widely used as feature to estimate the motion of the crowd. Using this feature, we can detect crowd density level or crowd events such as merge, split, walk, run, local dispersion and evacuation [138-141].

Cermeno et al. [142] used a combination of three different features (color, texture and shape) to train a multi-layer perceptron (MLP) to analyze simple crowd behavior: walking,

running, local dispersion, splitting, merging and evacuation. The results on the PETS 2009 dataset[26] showed an accuracy of 95 %.

Other methods propose dynamic texture methods such as the Mixture of Dynamic Texture (MDT) [143] and Local Binary Patterns (LBPTOP) [144] for recognizing crowd actions. Semantic approach has been also used for that purpose. We can state as an example the work of Rabiee et al. [145].

For more details about crowd representation and crowd behavior recognition and different categorization approach for these techniques, we refer the reader to these surveys [137,146,147]

In case of disjoint person/object action recognition, we are interested in the analysis of its behavior, understanding the interaction with its environment (person-object interaction) and/or other people. Such systems have many and various applications, such as change of speed or direction, falling, fight and abandonment of luggage.

In its simple way, we can consider such systems as an identification and classification problem. Therefore, all features (shape, motion and texture features) and classifiers used for object identification and classification can be used to detect and recognize these actions. However, because actions are a time varying process, the efficiency of these features are limited, which will decrease the accuracy of such systems. For that, we can consider this problem as a variable time data classification problem, and many features have been developed to achieve that. Chaudhry et al. [148] represented each frame of a video by a histogram of oriented optical flow (HOOF). First, they compute optical flow at every frame. Then, each flow vector is binned according to its primary angle from the horizontal axis and weighted according to its magnitude. The classification is then tested with leave-one-out, and the 1-Nearest Neighbor classifier is used with a special distance metric. The average recognition rate was found to be 94.4% on the Weizmann dataset[27].

Based on the previous work, Colque et al. [149] proposed a new feature, the Histograms of Optical Flow Orientation and Magnitude (HOFM), to solve the limitation of HOOF feature regarding the velocity of the moving objects. The classification step was performed using

[26] http://www.cvg.reading.ac.uk/PETS2009/a.html
[27] http://www.wisdom.weizmann.ac.il/%7Evision/SpaceTimeActions.html

nearest neighbor search. A comparison on the UCSD dataset[28] reported that this method outperforms the HOOF method.

Biologically-inspired features such as the HMAX (Hierarchical Model and X) [150,151] which has also been used for action recognition, is based on 'ventral stream' of visual cortex, it is one of the hypothesis used to explain how human neural system identify objects and recognize actions [152,153]. This system is imitated to identify objects [151] and to recognize actions [150]. The identification passes by different units to extract features. Spatiotemporal filters based on optical flow are applied on the first unit S1. The down-sampling is performed by computing local max in C1 unit. The template matching is performed in the S2 layer between C1 maps of the current frame and d1 stored templates (extracted during a training phase). The global max for S2 is computed in C2 layer. Finally, S3 units are obtained by considering 7 consecutive frames at a random location in the input sequence. As before, template matching is used to create S3 features followed by global max unit C3. The features extracted in C2 and C3 are fed to an SVM classifier, Fig.I.13.

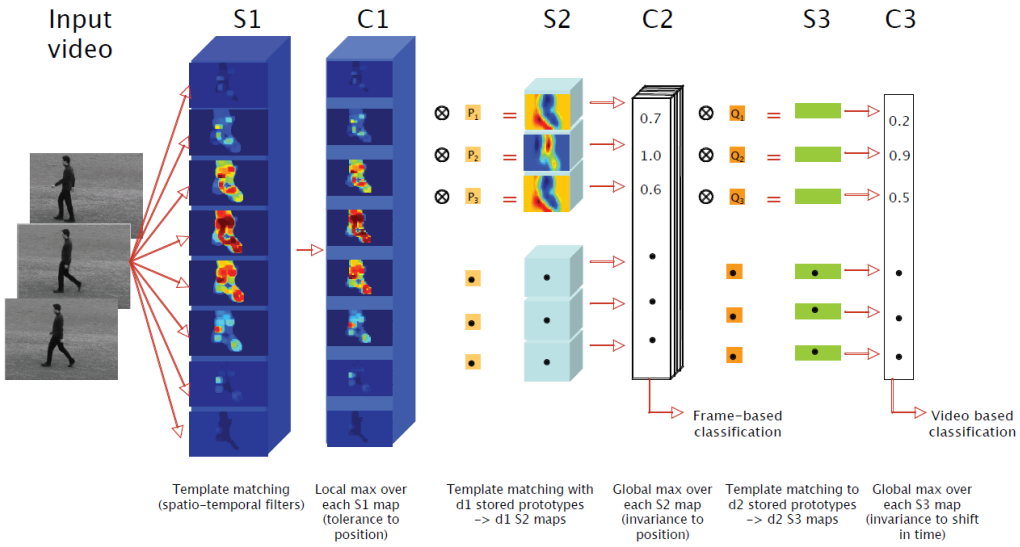


Fig.I.13. Sketch of the biologically inspired system for action recognition [150]

Other methods propose an extension of the previous features to spatio-temporal space, like the Hariss3D method proposed by Laptev and Lindeberg [154] and which is based on the Harris interest point detector [155]. Klaser et al. [156] proposed the HOG3D which is a generalization of the key HOG concepts to 3D by presenting the video as a spatio-temporal

[28] http://www.svcl.ucsd.edu/projects/anomaly/dataset.htm

volume. For the same purpose, Lowe [37] presented an extension of the SIFT descriptor to the 3D space in order to detect and recognize actions in video stream.

Dollar et al. [157] apply a spatiotemporal interest point detector to find local regions of interest in space and time, called "Cuboids". It serves as the substrate for behavior recognition. An extension of the SURF descriptor (Speeded Up Robust Features) to the spatio-temporal domain has been also proposed in [158].

As in the identification stage, the feature extraction step is followed by the classification step in which we can use all the classifiers mentioned above. However, other techniques have been developed in which a cell memory is incorporated to take into account the previous observation and to define the current action, e.g. the finite state machine FSM [127,159-162], dynamic time warping DTW[164-167], hidden Markov models HMM and coupled hidden Markov models [168-171], long-short term memory convolution neural network LSTM-CNN [91,172-175], Fully connected convolution neural network FC-CNN [176] and 3D convolution neuron network [177-180].

Only a few papers discuss the hardware implementation of action recognition techniques due to the memory bandwidth required for such systems. In the W^4 system [181], simple actions could be detected. The implementation was done on dual-Pentium and the system ran at 25Hz for low resolution image (320x280). Another recent implementation [182] is performed on Zynq-SoC FPGA for action recognition using a bag of words BoW[183] (HoG, HoF, histogram-oriented depth gradients HoDG) along with Harris corner detector and Hessian points[158]. In the classification step an SVM is used. Other real time implementations which have been conducted on GPU can be found in [179,184-186].

It should be noted that many competition and challenges for object and action recognition have been set and many of them still exist such as the ImageNet challenge[29], the annually ChaLearn Looking at People[30], THUMOS Challenge[31], HARL competition[32], Opportunity activity recognition challenge[33] and Kaggle competitions[34] (action recognition, image classification, digit classification with prizes that can exceed 1 million dollars!).

[29] https://www.kaggle.com/c/imagenet-object-detection-from-video-challenge
[30] http://chalearnlap.cvc.uab.es/challenge/23/description/
[31] http://www.thumos.info/
[32] http://liris.cnrs.fr/harl2012/
[33] http://opportunity-project.eu/challenge

The common dataset used for crowd detection and action recognition is presented in the table below.

Table I.4 Most used datasets for action recognition

Dataset	Description
KTH Dataset[35]	600 videos with resolution of 160×120 (192 training, 192 validation, 216 testing) for human action recognition, contains 6 classes: walking, jogging, running, boxing, hand waving, and hand clapping
Weizmann Dataset[36]	90 videos with resolution of 180×144 for human action recognition , it contains 10 classes : walk, run, jump, gallop sideways, bend, one-hand wave, two-hands wave, jump in place, jumping jack, skip
CMU Crowded Videos Dataset[37]	Video sequences of activities in crowded scenes to evaluate their proposed volumetric features. The videos are recorded using a hand-held camera. Each activity is performed by three to six subjects, resulting in 110 activities of interest. The videos are downscaled to 160×120 in resolution.
PETS 2009[38]	Eight videos for crowd analysis, which was introduced by the Performance Evaluation of Tracking and Surveillance Workshop (PETS 2009)
Hollywood1[39]	Dataset extracted from movies with eight different actions (answer phone, hug person, sit up, sit down, kiss, handshake, and stand up) The dataset consists of∼400 video sequences. Each sequence is about 50∼200 frames long with a resolution 240×500
Hollywood2[40]	An extension to the Hollywood1 dataset with up to twelve activities with a total of 600 K frames
UCF101[41]	UCF101 is an action recognition data set of realistic action videos, collected from YouTube, with 13320 videos having 101 action categories.
UCSD Anomaly Detection Dataset[42]	Two sets with 98 video sequences acquired with a stationary camera mounted at an elevation, the first set is for crowd action analysis, the second set is for pedestrian action recognition
ActivityNet[43]	A large-scale video benchmark for human activity understanding, with 203 activity classes in total of 849 video hours.

6. CONCLUSION

In this chapter, we have presented the state of the art of the different stages used to develop an intelligent video surveillance system. In the first part, we defined the intelligent video surveillance system and discussed the different hardware architecture used for real-time implementation of such system. We discussed the different motion detection techniques and their different hardware implementation achieved on FPGA and GPU circuits in the second part.

[34] https://www.kaggle.com/competitions
[35] http://www.nada.kth.se/cvap/actions/
[36] http://www.wisdom.weizmann.ac.il/%7Evision/SpaceTimeActions.html
[37] http://www.cs.cmu.edu/~yke/video/
[38] http://www.cvg.reading.ac.uk/PETS2009/a.html
[39] http://www.di.ens.fr/%7Elaptev/actions/
[40] http://www.di.ens.fr/%7Elaptev/actions/hollywood2/
[41] http://crcv.ucf.edu/data/UCF101.php
[42] http://www.svcl.ucsd.edu/projects/anomaly/dataset.htm
[43] http://activity-net.org/index.html

In the third part, we cited the different features and classifiers used for object identification as well as the different datasets used for benchmarking. Then we finished this part with different implementation found in the literature.

In the last part of this chapter, we reviewed the scene understanding techniques. First, we discussed the crowd behavior detection and the disjoint person/object behavior change detection, the different features and classifiers used for, and finally we cited the different datasets used for this purpose.

In the next chapter, we will discuss the different segmentation techniques used for motion detection purpose. The aim is to define the appropriate threshold method in order to fully automatize the video surveillance system.

REFERENCES

[1] G. Menezes and A. Silva-Filho, "Motion detection of vehicles based on FPGA," in Programmable Logic Conference (SPL), 2010 VI Southern, 2010, pp. 151-154.

[2] M. Gorgon, P. Pawlik, M. Jablonski, and J. Przybylo, "FPGA-based road traffic videodetector," in Digital System Design Architectures, Methods and Tools, 2007. DSD 2007. 10th Euromicro Conference on, 2007, pp. 412-419.

[3] T. Kryjak, M. Komorkiewicz, and M. Gorgon, "Real-time background generation and foreground object segmentation for high-definition colour video stream in FPGA device," Journal of Real-Time Image Processing, vol. 9, pp. 61-77, 2014.

[4] K. Liu, B. Ma, Q. Du, and G. Chen, "Fast motion detection from airborne videos using graphics processing unit," Journal of Applied Remote Sensing, vol. 6, pp. 061505-1-061505-14, 2012.

[5] C. Beleznai, D. Schreiber, and M. Rauter, "Pedestrian detection using GPU-accelerated multiple cue computation," in Computer Vision and Pattern Recognition Workshops (CVPRW), 2011 IEEE Computer Society Conference on, 2011, pp. 58-65.

[6] C. Sanchez-Ferreira, J. Mori, and C. Llanos, "Background subtraction algorithm for moving object detection in FPGA," in Programmable Logic (SPL), 2012 VIII Southern Conference on, 2012, pp. 1-6.

[7] Y. Li, K. Gai, M. Qiu, W. Dai, and M. Liu, "Adaptive human detection approach using FPGA-based parallel architecture in reconfigurable hardware," Concurrency and Computation: Practice and Experience, vol. 29, 2017.

[8] S. Singh, A. S. Mandal, C. Shekhar, and A. Vohra, "Memory Efficient VLSI Implementation of Real-Time Motion Detection System Using FPGA Platform," Journal of Imaging, vol. 3, p. 20, 2017.

[9] T. Zhang, H. Wu, A. Borst, K. Kuhnlenz, and M. Buss, "An FPGA implementation of insect-inspired motion detector for high-speed vision systems," in Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on, 2008, pp. 335-340.

[10] L. Bako, S. Hajdu, S.-T. Brassai, F. Morgan, and C. Enachescu, "Embedded Implementation of a Real-Time Motion Estimation Method in Video Sequences," Procedia Technology, vol. 22, pp. 897-904, 2016.

[11] W. Hu, T. Tan, L. Wang, and S. Maybank, "A survey on visual surveillance of object motion and behaviors," IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), vol. 34, pp. 334-352, 2004.

[12] M. Paul, S. M. Haque, and S. Chakraborty, "Human detection in surveillance videos and its applications-a review," EURASIP Journal on Advances in Signal Processing, vol. 2013, p. 176, 2013.

[13] P. Yan, S. M. Khan, and M. Shah, "3d model based object class detection in an arbitrary view," in Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on, 2007, pp. 1-6.

[14] D. Huber, A. Kapuria, R. Donamukkala, and M. Hebert, "Parts-based 3d object classification," in Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on, 2004, pp. II-II.

[15] S. Biasotti, D. Giorgi, S. Marini, M. Spagnuolo, and B. Falcidieno, "A comparison framework for 3d object classification methods," in International Workshop on Multimedia Content Representation, Classification and Security, 2006, pp. 314-321.

[16] R. Layne, T. M. Hospedales, S. Gong, and Q. Mary, "Person Re-identification by Attributes," in Bmvc, 2012, p. 8.

[17] S. Liu, Z. Song, G. Liu, C. Xu, H. Lu, and S. Yan, "Street-to-shop: Cross-scenario clothing retrieval via parts alignment and auxiliary set," in Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, 2012, pp. 3330-3337.

[18] M. Frikha, E. Fendri, and M. Hammami, "Semantic attributes for people's appearance description: an appearance modality for video surveillance applications," Journal of Electronic Imaging, vol. 26, p. 051405, 2017.

[19] H.-Y. Lin and J.-Y. Wei, "A street scene surveillance system for moving object detection, tracking and classification," in Intelligent Vehicles Symposium, 2007 IEEE, 2007, pp. 1077-1082.

[20] A. J. Lipton, H. Fujiyoshi, and R. S. Patil, "Moving target classification and tracking from real-time video," in Applications of Computer Vision, 1998. WACV'98. Proceedings., Fourth IEEE Workshop on, 1998, pp. 8-14.

[21] R. T. Collins, A. J. Lipton, T. Kanade, H. Fujiyoshi, D. Duggins, Y. Tsin, et al., "A system for video surveillance and monitoring," VSAM final report, pp. 1-68, 2000.

[22] R. N. Hota, V. Venkoparao, and A. Rajagopal, "Shape based object classification for automated video surveillance with feature selection," in Information Technology,(ICIT 2007). 10th International Conference on, 2007, pp. 97-99.

[23] H. Van der Werff and F. Van der Meer, "Shape-based classification of spectrally identical objects," ISPRS Journal of Photogrammetry and Remote Sensing, vol. 63, pp. 251-258, 2008.

[24] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-based object tracking," IEEE Transactions on pattern analysis and machine intelligence, vol. 25, pp. 564-577, 2003.

[25] J. Zhang, "Motor vehicle occupant detection system employing ellipse shape models and bayesian classification," ed: Google Patents, 2002.

[26] C. Teutsch, D. Berndt, E. Trostmann, and M. Weber, "Real-time detection of elliptic shapes for automated object recognition and object tracking," Proc. of Machine Vision Applications in Industrial Inspection XIV (Electronic Imaging 2006, pp. 171-179, 2006.

[27] R. Cutler and L. S. Davis, "Robust real-time periodic motion detection, analysis, and applications," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, pp. 781-796, 2000.

[28] O. Javed and M. Shah, "Tracking and object classification for automated surveillance," in European Conference on Computer Vision, 2002, pp. 343-357.

[29] A. F. Bobick and J. W. Davis, "The recognition of human movement using temporal templates," IEEE Transactions on pattern analysis and machine intelligence, vol. 23, pp. 257-267, 2001.

[30] Y. Ran, I. Weiss, Q. Zheng, and L. S. Davis, "Pedestrian detection via periodic motion analysis," International Journal of Computer Vision, vol. 71, pp. 143-160, 2007.

[31] C. Papageorgiou and T. Poggio, "A trainable system for object detection," International Journal of Computer Vision, vol. 38, pp. 15-33, 2000.

[32] M. Oren, C. Papageorgiou, P. Sinha, E. Osuna, and T. Poggio, "Pedestrian detection using wavelet templates," in Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on, 1997, pp. 193-199.

[33] A. Apatean, A. ROGOZAN, S. EMERICH, and A. BENSRHAIR, "Wavelets as features for objects recognition," Acta Tehnica Napocensis-Electronics and Telecommunications, vol. 49, pp. 23-26, 2008.

[34] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," IEEE transactions on pattern analysis and machine intelligence, vol. 24, pp. 509-522, 2002.

[35] S. G. Salve and K. C. Jondhale, "Shape matching and object recognition using shape contexts," in Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on, 2010, pp. 471-474.

[36] M. Wang, J.-Q. Wang, H. Qiao, and X.-B. Cao, "Improved shape context algorithm for online fast recognition-an application in pedestrian detection from a moving vehicle," in Intelligent Vehicles Symposium, 2009 IEEE, 2009, pp. 1260-1264.

[37] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," International journal of computer vision, vol. 60, pp. 91-110, 2004.

[38] K. Mikolajczyk, C. Schmid, and A. Zisserman, "Human detection based on a probabilistic assembly of robust part detectors," Computer Vision-ECCV 2004, pp. 69-82, 2004.

[39] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, 2005, pp. 886-893.

[40] T. B. Moeslund, A. Hilton, and V. Krüger, "A survey of advances in vision-based human motion capture and analysis," Computer vision and image understanding, vol. 104, pp. 90-126, 2006.

[41] P. Negri, X. Clady, and L. Prevost, "Benchmarking haar and histograms of oriented gradients features applied to vehicle detection," in ICINCO-RA (1), 2007, pp. 359-364.

[42] T. Ojala, M. Pietikäinen, and D. Harwood, "A comparative study of texture measures with classification based on featured distributions," Pattern recognition, vol. 29, pp. 51-59, 1996.

[43] L. Wang and D.-C. He, "Texture classification using texture spectrum," Pattern Recognition, vol. 23, pp. 905-910, 1990.

[44] M. Heikkilä, M. Pietikäinen, and J. Heikkilä, "A texture-based method for detecting moving objects," in Bmvc, 2004, pp. 1-10.

[45] J. Trefný and J. Matas, "Extended set of local binary patterns for rapid object detection," in Computer Vision Winter Workshop, 2010, pp. 1-7.

[46] D. Xia, H. Sun, and Z. Shen, "Real-time infrared pedestrian detection based on multi-block LBP," in Computer Application and System Modeling (ICCASM), 2010 International Conference on, 2010, pp. V12-139-V12-142.

[47] A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman, "Multiple kernels for object detection," in Computer Vision, 2009 IEEE 12th International Conference on, 2009, pp. 606-613.

[48] T. Deselaers and V. Ferrari, "Global and efficient self-similarity for object classification and detection," in Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, 2010, pp. 1633-1640.

[49] C. BenAbdelkader, R. Cutler, H. Nanda, and L. Davis, "Eigengait: Motion-based recognition of people using image self-similarity," in Audio-and Video-Based Biometric Person Authentication, 2001, pp. 284-294.

[50] C. Silva, T. Bouwmans, and C. Frélicot, "An eXtended center-symmetric local binary pattern for background modeling and subtraction in videos," in International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, VISAPP 2015, 2015.

[51] T. Bouwmans, C. Silva, C. Marghes, M. S. Zitouni, H. Bhaskar, and C. Frelicot, "On the Role and the Importance of Features for Background Modeling and Foreground Detection," arXiv preprint arXiv:1611.09099, 2016.

[52] X. Wu, V. Kumar, J. R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, et al., "Top 10 algorithms in data mining," Knowledge and information systems, vol. 14, pp. 1-37, 2008.

[53] J. R. Quinlan, C4. 5: programs for machine learning: Elsevier, 2014.

[54] A. K. Jain and R. C. Dubes, Algorithms for clustering data: Prentice-Hall, Inc., 1988.

[55] R. M. Gray and D. L. Neuhoff, "Quantization," IEEE transactions on information theory, vol. 44, pp. 2325-2383, 1998.

[56] V. Vapnik, The nature of statistical learning theory: Springer science & business media, 2013.

[57] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," in Proc. 20th int. conf. very large data bases, VLDB, 1994, pp. 487-499.

[58] G. McLachlan and T. Krishnan, The EM algorithm and extensions vol. 382: John Wiley & Sons, 2007.

[59] G. McLachlan and D. Peel, Finite mixture models: John Wiley & Sons, 2004.

[60] S. Brin and L. Page, "Reprint of: The anatomy of a large-scale hypertextual web search engine," Computer networks, vol. 56, pp. 3825-3833, 2012.

[61] Y. Freund and R. E. Schapire, "A desicion-theoretic generalization of on-line learning and an application to boosting," in European conference on computational learning theory, 1995, pp. 23-37.

[62] E. Fix and J. L. Hodges Jr, "Discriminatory analysis-nonparametric discrimination: consistency properties," California Univ Berkeley1951.

[63] P.-N. Tan, Introduction to data mining: Pearson Education India, 2006.

[64] P. Domingos and M. Pazzani, "On the optimality of the simple Bayesian classifier under zero-one loss," Machine learning, vol. 29, pp. 103-130, 1997.

[65] D. J. Hand and K. Yu, "Idiot's Bayes—not so stupid after all?," International statistical review, vol. 69, pp. 385-398, 2001.

[66] W. Y. Loh, "Classification and regression trees," Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, vol. 1, pp. 14-23, 2011.

[67] D. A. Freedman, Statistical models: theory and practice: cambridge university press, 2009.

[68] X. Yan and X. Su, Linear regression analysis: theory and computing: World Scientific, 2009.

[69] S.-I. Oh and H.-B. Kang, "Object detection and classification by decision-level fusion for intelligent vehicle systems," Sensors, vol. 17, p. 207, 2017.

[70] S. H. Walker and D. B. Duncan, "Estimation of the probability of an event as a function of several independent variables," Biometrika, vol. 54, pp. 167-179, 1967.

[71] J. Kim, J. Lee, C. Lee, E. Park, J. Kim, H. Kim, et al., "Optimal Feature Selection for Pedestrian Detection based on Logistic Regression Analysis," in Systems, Man, and Cybernetics (SMC), 2013 IEEE International Conference on, 2013, pp. 239-242.

[72] A. Khellal, H. Ma, and Q. Fei, "Pedestrian Classification and Detection in Far Infrared Images," in International Conference on Intelligent Robotics and Applications, 2015, pp. 511-522.

[73] B. Schölkopf, A. Smola, and K.-R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," Neural computation, vol. 10, pp. 1299-1319, 1998.

[74] M. Welling, "Kernel principal components analysis," Advances in Neura L Information Processing Systems, vol. 15, pp. 70-72, 2003.

[75] Y.-h. Liang, Z.-y. Wang, S. Guo, X.-w. Xu, and X.-y. Cao, "Pedestrian detection using kpca and fld algorithms," in Automation and Logistics, 2007 IEEE International Conference on, 2007, pp. 1572-1575.

[76] K. Adamiak, P. Duch, and K. Ślot, "Object Classification using Support Vector Machines with Kernel-based Data Preprocessing," Image Processing & Communications, vol. 21, pp. 45-53, 2016.

[77] B. Qi, V. John, Z. Liu, and S. Mita, "Use of sparse representation for pedestrian detection in thermal images," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2014, pp. 274-280.

[78] C.-H. Zheng, W.-J. Pei, Q. Yan, and Y.-W. Chong, "Pedestrian detection based on gradient and texture feature integration," Neurocomputing, vol. 228, pp. 71-78, 2017.

[79] P. J. Werbos, "Beyond regression: New tools for prediction and analysis in the behavioral sciences," Doctoral Dissertation, Applied Mathematics, Harvard University, MA, 1974.

[80] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," California Univ San Diego La Jolla Inst for Cognitive Science1985.

[81] R. Gouiaa and J. Meunier, "Learning cast shadow appearance for human posture recognition," Pattern Recognition Letters, vol. 97, pp. 54-60, 2017.

[82] J. L. Elman, "Finding structure in time," Cognitive science, vol. 14, pp. 179-211, 1990.

[83] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proceedings of the IEEE, vol. 86, pp. 2278-2324, 1998.

[84] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2014, pp. 580-587.

[85] R. Girshick, "Fast r-cnn," in Proceedings of the IEEE international conference on computer vision, 2015, pp. 1440-1448.

[86] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," IEEE transactions on pattern analysis and machine intelligence, vol. 39, pp. 1137-1149, 2017.

[87] [87] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," arXiv preprint arXiv:1703.06870, 2017.

[88] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural computation, vol. 9, pp. 1735-1780, 1997.

[89] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, et al., "Long-term recurrent convolutional networks for visual recognition and description," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 2625-2634.

[90] E. Tsironi, P. Barros, C. Weber, and S. Wermter, "An analysis of Convolutional Long Short-Term Memory Recurrent Neural Networks for gesture recognition," Neurocomputing, 2017.

[91] N. Srivastava, E. Mansimov, and R. Salakhudinov, "Unsupervised learning of video representations using lstms," in International Conference on Machine Learning, 2015, pp. 843-852.

[92] J. R. Medel and A. Savakis, "Anomaly detection in video using predictive convolutional long short-term memory networks," arXiv preprint arXiv:1612.00390, 2016.

[93] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in Advances in neural information processing systems, 2007, pp. 153-160.

[94] A. Krizhevsky, "One weird trick for parallelizing convolutional neural networks," arXiv preprint arXiv:1404.5997, 2014.

[95] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv preprint arXiv:1409.1556, 2014.

[96] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, et al., "Going deeper with convolutions," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 1-9.

[97] R. Caruana and A. Niculescu-Mizil, "An empirical comparison of supervised learning algorithms," in Proceedings of the 23rd international conference on Machine learning, 2006, pp. 161-168.

[98] N. Settouti, M. E. A. Bechar, and M. A. Chikh, "Statistical comparisons of the top 10 algorithms in data mining for classification task," Int. J. Interact. Multimedia Artif. Intell., Special Issue Artif. Intell. Underpinning, vol. 4, pp. 46-51, 2016.

[99] C.-C. Chang and C.-J. Lin, "LIBSVM: a library for support vector machines," ACM transactions on intelligent systems and technology (TIST), vol. 2, p. 27, 2011.

[100] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal, Data Mining: Practical machine learning tools and techniques: Morgan Kaufmann, 2016.

[101] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: an update," ACM SIGKDD explorations newsletter, vol. 11, pp. 10-18, 2009.

[102] C. R. Souza, "The Accord .NET Framework," ht tp://accord-framework. net. São Carlos, Brazil, 2014.

[103] B. Meus,T. Kryjak and M. Gorgon, " Embedded vision system for pedestrian detection based on HOG+SVM and use of motion information implemented in Zynq heterogeneous device," 21st Conference SPA 2017 Signal Processing: Algorithms, Architectures, Arrangements, and Applications, At Poznań, Poland, 2017.

[104] C. Kyrkou and T. Theocharides, "Accelerating object detection via a visual-feature-directed search cascade: algorithm and field programmable gate array implementation," Journal of Electronic Imaging, vol. 25, pp. 041013-041013, 2016.

[105] M. Hiromoto, H. Sugano, and R. Miyamoto, "Partially parallel architecture for adaboost-based detection with haar-like features," IEEE Transactions on Circuits and Systems for Video Technology, vol. 19, pp. 41-52, 2009.

[106] T. Kryjak, M. Komorkiewicz, and M. Gorgon, "FPGA implementation of real-time head-shoulder detection using local binary patterns, SVM and foreground object

detection," in Design and Architectures for Signal and Image Processing (DASIP), 2012 Conference on, 2012, pp. 1-8.

[107] T. Kańka, T. Kryjak, and M. Gorgon, "FPGA Implementation of Multi-scale Pedestrian Detection in Thermal Images," Image Processing & Communications, vol. 21, pp. 55-67, 2016.

[108] H. Xiao, H. Song, W. He, and K. Yuan, "Real-time shape and pedestrian detection with FPGA," in Mechatronics and Automation (ICMA), 2015 IEEE International Conference on, 2015, pp. 2381-2386.

[109] R. Walczyk, A. Armitage, and T. D. Binnie, "An embedded real-time pedestrian detection system using an infrared camera," 2009.

[110] M. Drożdż and T. Kryjak, "FPGA Implementation of Multi-scale Face Detection Using HOG Features and SVM Classifier," Image Processing & Communications, vol. 21, pp. 27-44, 2016.

[111] S. Heymann, K. Müller, A. Smolic, B. Froehlich, and T. Wiegand, "SIFT implementation and optimization for general-purpose GPU," 2007.

[112] V. Prisacariu and I. Reid, "fastHOG-a real-time GPU implementation of HOG," Department of Engineering Science, vol. 2310, 2009.

[113] [110] B. Wu and R. Nevatia, "Cluster boosted tree classifier for multi-view, multi-pose object detection," in Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on, 2007, pp. 1-8.

[114] B. Wu and R. Nevatia, "Detection of multiple, partially occluded humans in a single image by bayesian combination of edgelet part detectors," in Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on, 2005, pp. 90-97.

[115] G. Overett, L. Petersson, N. Brewer, L. Andersson, and N. Pettersson, "A new pedestrian dataset for supervised learning," in Intelligent Vehicles Symposium, 2008 IEEE, 2008, pp. 373-378.

[116] S. Silberstein, D. Levi, V. Kogan, and R. Gazit, "Vision-based pedestrian detection for rear-view cameras," in Intelligent Vehicles Symposium Proceedings, 2014 IEEE, 2014, pp. 853-860.

[117] D. Levi and S. Silberstein, "Tracking and motion cues for rear-view pedestrian detection," in Intelligent Transportation Systems (ITSC), 2015 IEEE 18th International Conference on, 2015, pp. 664-671.

[118] A. González, Z. Fang, Y. Socarras, J. Serrat, D. Vázquez, J. Xu, et al., "Pedestrian detection at day/night time with visible and FIR cameras: A comparison," Sensors, vol. 16, p. 820, 2016.

[119] M. Enzweiler and D. M. Gavrila, "Monocular pedestrian detection: Survey and experiments," IEEE transactions on pattern analysis and machine intelligence, vol. 31, pp. 2179-2195, 2009.

[120] S. M. Bileschi, "StreetScenes: Towards scene understanding in still images," MASSACHUSETTS INST OF TECH CAMBRIDGE2006.

[121] P. Dollar, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: An evaluation of the state of the art," IEEE transactions on pattern analysis and machine intelligence, vol. 34, pp. 743-761, 2012.

[122] A. Ess, B. Leibe, K. Schindler, and L. Van Gool, "A mobile vision system for robust multi-person tracking," in Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on, 2008, pp. 1-8.

[123] C. Wojek, S. Walk, and B. Schiele, "Multi-cue onboard pedestrian detection," in Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, 2009, pp. 794-801.

[124] T. Li, H. Chang, M. Wang, B. Ni, R. Hong, and S. Yan, "Crowded scene analysis: A survey," IEEE transactions on circuits and systems for video technology, vol. 25, pp. 367-386, 2015.

[125] N. T. Siebel and S. Maybank, "The advisor visual surveillance system," in ECCV 2004 workshop applications of computer vision (ACV), 2004.

[126] F. Cupillard, A. Avanzi, F. Bremond, and M. Thonnat, "Video understanding for metro surveillance," in Networking, Sensing and Control, 2004 IEEE International Conference on, 2004, pp. 186-191.

[127] T. Arikuma and Y. Mochizuki, "Intelligent multimedia surveillance system for safer cities," APSIPA Transactions on Signal and Information Processing, vol. 5, 2016.

[128] T. Arikuma, A. Jain, K. Koyama, K. W. Woo, T. Kawamata, and K. Yamada, "Content Based Video Quality Control for Wide-area Video Surveillance Systems."

[129] T. Arikuma, K. Koyama, T. Kitano, N. Shiraishi, Y. Nagai, and T. Kawamata, "Analysis control middleware for large-scale video surveillance," in Advanced Video

and Signal Based Surveillance (AVSS), 2013 10th IEEE International Conference on, 2013, pp. 294-299.

[130] P. BERNAS, G. NEE, and P. DRABCZUK, "Peaceful Monitoring of Crowds."

[131] B. E. Moore, S. Ali, R. Mehran, and M. Shah, "Visual crowd surveillance through a hydrodynamics lens," Communications of the ACM, vol. 54, pp. 64-73, 2011.

[132] W.-Y. Ren, G.-H. Li, J. Chen, and H.-Z. Liang, "Abnormal crowd behavior detection using behavior entropy model," in Wavelet Analysis and Pattern Recognition (ICWAPR), 2012 International Conference on, 2012, pp. 212-221.

[133] N. Ihaddadene and C. Djeraba, "Real-time crowd motion analysis," in Pattern Recognition, 2008. ICPR 2008. 19th International Conference on, 2008, pp. 1-4.

[134] D. Helbing and P. Molnar, "Social force model for pedestrian dynamics," Physical review E, vol. 51, p. 4282, 1995.

[135] R. Mehran, A. Oyama, and M. Shah, "Abnormal crowd behavior detection using social force model," in Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, 2009, pp. 935-942.

[136] J. Zhao, Y. Xu, X. Yang, and Q. Yan, "Crowd instability analysis using velocity-field based social force model," in Visual Communications and Image Processing (VCIP), 2011 IEEE, 2011, pp. 1-4.

[137] H. Jo, K. Chug, and R. J. Sethi, "A review of physics-based methods for group and crowd analysis in computer vision," Journal of Postdoctoral Research, vol. 1, pp. 4-7, 2013.

[138] Y. Benabbas, N. Ihaddadene, and C. Djeraba, "Motion pattern extraction and event detection for automatic visual surveillance," EURASIP Journal on Image and Video Processing, vol. 2011, p. 163682, 2011.

[139] I. Saleemi, L. Hartung, and M. Shah, "Scene understanding by statistical modeling of motion patterns," in Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, 2010, pp. 2069-2076.

[140] L. Song, F. Jiang, Z. Shi, and A. K. Katsaggelos, "Understanding dynamic scenes by hierarchical motion pattern mining," in Multimedia and Expo (ICME), 2011 IEEE International Conference on, 2011, pp. 1-6.

[141] S. Wu, H.-S. Wong, and Z. Yu, "A Bayesian model for crowd escape behavior detection," IEEE transactions on circuits and systems for video technology, vol. 24, pp. 85-98, 2014.

[142] E. Cermeno, S. Mallor, and J. A. Sigüenza, "Learning crowd behavior for event recognition," in Performance Evaluation of Tracking and Surveillance (PETS), 2013 IEEE International Workshop on, 2013, pp. 1-5.

[143] W. Li, V. Mahadevan, and N. Vasconcelos, "Anomaly detection and localization in crowded scenes," IEEE transactions on pattern analysis and machine intelligence, vol. 36, pp. 18-32, 2014.

[144] J. Xu, S. Denman, S. Sridharan, C. Fookes, and R. Rana, "Dynamic texture reconstruction from sparse codes for unusual event detection in crowded scenes," in Proceedings of the 2011 joint ACM workshop on Modeling and representing events, 2011, pp. 25-30.

[145] H. Rabiee, J. Haddadnia, and H. Mousavi, "Crowd behavior representation: an attribute-based approach," SpringerPlus, vol. 5, p. 1179, 2016.

[146] V. J. Kok, M. K. Lim, and C. S. Chan, "Crowd behavior analysis: A review where physics meets biology," Neurocomputing, vol. 177, pp. 342-362, 2016.

[147] C. J. Dhamsania and T. V. Ratanpara, "A survey on Human action recognition from videos," in Green Engineering and Technologies (IC-GET), 2016 Online International Conference on, 2016, pp. 1-5.

[148] R. Chaudhry, A. Ravichandran, G. Hager, and R. Vidal, "Histograms of oriented optical flow and binet-cauchy kernels on nonlinear dynamical systems for the recognition of human actions," in Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, 2009, pp. 1932-1939.

[149] R. V. H. M. Colque, C. Caetano, M. T. L. de Andrade, and W. R. Schwartz, "Histograms of Optical Flow Orientation and Magnitude and Entropy to Detect Anomalous Events in Videos," IEEE Transactions on Circuits and Systems for Video Technology, vol. 27, pp. 673-682, 2017.

[150] H. Jhuang, T. Serre, L. Wolf, and T. Poggio, "A biologically inspired system for action recognition," in Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on, 2007, pp. 1-8.

[151] T. Serre, L. Wolf, S. Bileschi, M. Riesenhuber, and T. Poggio, "Robust object recognition with cortex-like mechanisms," IEEE transactions on pattern analysis and machine intelligence, vol. 29, pp. 411-426, 2007.

[152] M. A. Goodale and A. D. Milner, "Separate visual pathways for perception and action," Trends in neurosciences, vol. 15, pp. 20-25, 1992.

[153] M. Riesenhuber and T. Poggio, "Hierarchical models of object recognition in cortex," Nature neuroscience, vol. 2, pp. 1019-1025, 1999.

[154] I. Laptev, "On space-time interest points," International journal of computer vision, vol. 64, pp. 107-123, 2005.

[155] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, et al., "A comparison of affine region detectors," International journal of computer vision, vol. 65, pp. 43-72, 2005.

[156] A. Klaser, M. Marszałek, and C. Schmid, "A spatio-temporal descriptor based on 3d-gradients," in BMVC 2008-19th British Machine Vision Conference, 2008, pp. 275: 1-10.

[157] P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie, "Behavior recognition via sparse spatio-temporal features," in Visual Surveillance and Performance Evaluation of Tracking and Surveillance, 2005. 2nd Joint IEEE International Workshop on, 2005, pp. 65-72.

[158] G. Willems, T. Tuytelaars, and L. Van Gool, "An efficient dense and scale-invariant spatio-temporal interest point detector," Computer Vision–ECCV 2008, pp. 650-663, 2008.

[159] Y. Han, S.-L. Chung and S.-F. Su, "Automatic Action Segmentation and Continuous Recognition for Basic Indoor Actions Based on Kinect Pose Streams," International Conference on Systems, Man, and Cybernetics (SMC)Banff Center, Banff, Canada, October 5-8, 2017.

[160] F. Bremond and G. Medioni, "Scenario recognition in airborne video imagery," in Proc. Int. Workshop Interpretation of Visual Motion, 1998, pp. 57-64.

[161] N. Noorit and N. Suvonvorn, "Human activity recognition from basic actions using finite state machine," in Proceedings of the First International Conference on Advanced Data and Information Engineering (DaEng-2013), 2014, pp. 379-386.

[162] D. Dudzński, T. Kryjak, and Z. Mikrut, "Human action recognition using simple geometric features and a finite state machine," Image Processing & Communications, vol. 18, pp. 49-60, 2013.

[163] K. Sehairi, C. Benbouchama, and F. Chouireb, "Real-time implementation of human action recognition system based on motion analysis," in Artificial Intelligence and Computer Vision, ed: Springer, 2017, pp. 143-164.

[164] Z. Z. Htike, S. Egerton, and K. Y. Chow, "Monocular viewpoint invariant human activity recognition," in Robotics, Automation and Mechatronics (RAM), 2011 IEEE Conference on, 2011, pp. 18-23.

[165] H. Cheng, Z. Dai, Z. Liu, and Y. Zhao, "An image-to-class dynamic time warping approach for both 3D static and trajectory hand gesture recognition," Pattern Recognition, vol. 55, pp. 137-147, 2016.

[166] C. H. Pham, Q. K. Le, and T. H. Le, "Human Action Recognition Using Dynamic Time Warping and Voting Algorithm (1)," 2014.

[167] V. Tripathi, S. Agarwal, A. Mittal, and D. Gangodkar, "Improved Dynamic Time Warping Based Approach for Activity Recognition," in Proceedings of the 5th International Conference on Frontiers in Intelligent Computing: Theory and Applications, 2017, pp. 161-167.

[168] M. Brand, N. Oliver, and A. Pentland, "Coupled hidden Markov models for complex action recognition," in Computer vision and pattern recognition, 1997. proceedings., 1997 ieee computer society conference on, 1997, pp. 994-999.

[169] R. Cilla, M. A. Patricio, J. García, A. Berlanga, and J. M. Molina, "Recognizing human activities from sensors using hidden markov models constructed by feature selection techniques," Algorithms, vol. 2, pp. 282-300, 2009.

[170] W. Huang, J. Zhang, and Z. Liu, "Activity recognition based on hidden Markov models," Knowledge Science, Engineering and Management, pp. 532-537, 2007.

[171] M. H. Kabir, M. R. Hoque, K. Thapa, and S.-H. Yang, "Two-layer hidden markov model for human activity recognition in home environments," International Journal of Distributed Sensor Networks, vol. 12, p. 4560365, 2016.

[172] C. Li, P. Wang, S. Wang, Y. Hou, and W. Li, "Skeleton-based action recognition using LSTM and CNN," in Multimedia & Expo Workshops (ICMEW), 2017 IEEE International Conference on, 2017, pp. 585-590.

[173] J. Yue-Hei Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, "Beyond short snippets: Deep networks for video classification," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 4694-4702.

[174] Z. Wu, X. Wang, Y.-G. Jiang, H. Ye, and X. Xue, "Modeling spatial-temporal clues in a hybrid deep learning framework for video classification," in Proceedings of the 23rd ACM international conference on Multimedia, 2015, pp. 461-470.

[175] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A search space odyssey," IEEE transactions on neural networks and learning systems, 2017.

[176] X. Yang, P. Molchanov, and J. Kautz, "Multilayer and multimodal fusion of deep neural networks for video classification," in Proceedings of the 2016 ACM on Multimedia Conference, 2016, pp. 978-987.

[177] M. Baccouche, F. Mamalet, C. Wolf, C. Garcia, and A. Baskurt, "Sequential deep learning for human action recognition," in International Workshop on Human Behavior Understanding, 2011, pp. 29-39.

[178] S. Ji, W. Xu, M. Yang, and K. Yu, "3D convolutional neural networks for human action recognition," IEEE transactions on pattern analysis and machine intelligence, vol. 35, pp. 221-231, 2013.

[179] M. S. Rajeswar, A. R. Sankar, V. N. Balasubramaniam, and C. Sudheer, "Scaling up the training of deep CNNs for human action recognition," in Parallel and Distributed Processing Symposium Workshop (IPDPSW), 2015 IEEE International, 2015, pp. 1172-1177.

[180] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks," in Proceedings of the IEEE international conference on computer vision, 2015, pp. 4489-4497.

[181] I. Haritaoglu, D. Harwood, and L. S. Davis, "W/sup 4: real-time surveillance of people and their activities," IEEE Transactions on pattern analysis and machine intelligence, vol. 22, pp. 809-830, 2000.

[182] A. Safaei and Q. J. Wu, "System-Level Design for Human Action Recognition in 3D Scenes."

[183] S. Hadfield, K. Lebeda, and R. Bowden, "Hollywood 3D: What are the best 3D features for Action Recognition?," International Journal of Computer Vision, vol. 121, pp. 95-110, 2017.

[184] X. Ma, J. R. Borbon, W. Najjar, and A. K. Roy-Chowdhury, "Optimizing hardware design for human action recognition," in Field Programmable Logic and Applications (FPL), 2016 26th International Conference on, 2016, pp. 1-11.

[185] B. Zhang, L. Wang, Z. Wang, Y. Qiao, and H. Wang, "Real-time action recognition with enhanced motion vector CNNs," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2718-2726.

[186] M. Bayazit, A. Couture-Beil, and G. Mori, "Real-time Motion-based Gesture Recognition Using the GPU," in MVA, 2009, pp. 9-12.

Chapter II

Threshold Algorithms for Motion Detection

One who follows the crowd will surpass solitary individual and together with the crowd, ‘venture beyond places’ where no lone individual is capable of venturing to. Emergent behavior phenomenon

1. INTRODUCTION

Motion detection is the most important stage in video surveillance systems. Good results of this stage are not justified only by the choice of the method but also by the good segmentation and the adaptation to changes in luminance. The good choice of an automatic threshold will give the best segmentation and good detection results. The thresholding operation is a simple method that consists of separating the objects from background. But various factors, such as non-stationary and correlated noise, ambient illumination, busyness of gray levels within the object and its background, inadequate contrast and object size which is not commensurate with the scene, complicate this operation [1]. Basically, there are two different approaches for thresholding: global thresholding and local thresholding. In global thresholding, a single threshold for all the image pixels is used. In local thresholding, the image is divided into smaller sub-images and the threshold for each sub-image depends on local properties of the point or on its position [2]. A wide variety of image thresholding algorithms exists in literature [1,2,3,4], we can categorize them in five categories [3]: Histogram-based methods, Cluster based methods, Entropy-based methods, Object attribute based methods and local based ones.

In this chapter, an attempt to choose the best thresholding method for object motion detection in indoor and outdoor areas has been done by comparing different thresholding methods: the Otsu's method [5], the iterative selection (ISODATA Iterative Self-Organizing Data Analysis Technique) [6], Kapur's Entropy thresholding [7], Ramesh's threshold [8], Tsai's threshold [9] and GMM threshold method [10]. We have tested all these threshold methods on different motion detection algorithms and for different indoor/outdoor scenes in order to find the method that match the best threshold algorithm in the aim of obtaining the best results.

For motion detection, there are three conventional approaches for moving object detection classified in [11] as: temporal differencing, background subtraction, and optical flow. Temporal differencing is very adaptive to dynamic environments, but generally does a poor job of extracting all relevant feature pixels. Background subtraction provides the most complete feature data, but is extremely sensitive to dynamic scene changes due to lighting and extraneous events. Optical flow can be used to detect independently moving objects even in the presence of camera motion; however, most optical flow computation methods are computationally complex, and cannot be applied to full-frame video streams in real-time without specialized hardware.

In this work, we have chosen three motion detection methods (frame differencing and two background subtraction techniques): the delta frame method [12- 15], running average filter (recursive background detection) [16-19] and hybrid motion detection method [11,20].

To evaluate the performance of each combination, we have used three Pixel-based metrics [21,22]: the Percentage of Correct Classification (PCC), the Jaccard Coefficient (JC) and the Yule Coefficient (YC).

This chapter is organized as follows. In the next section a review of thresholding methods is presented. It is followed, in Section 3, by a review of motion detection algorithms. In section 4, the application of different threshold methods on these different motion detection algorithms for many scenes is illustrated. Finally, a common discussion on these results and a pixel-based evaluation is presented before concluding the chapter.

2. THRESHOLD METHODS

Thresholding is a simple, fast and easy method for image segmentation which is successfully used in a wide spectrum of practical applications of vision systems [4]. Typically a black and white binary image is obtained after this operation. In thresholding we transform an input image ζ into a binary image Ψ. If $\zeta(x,y)$ is the result of motion segmentation in gray-level of the pixel (x,y) in ζ, then the corresponding values in Ψ are:

$$\Psi(x,y)=\begin{cases}0 & \text{if } \zeta(x,y)<Th \quad \text{for static background}\\ 1 & \text{otherwise} \quad\quad \text{for moving object}\end{cases} \tag{II.1}$$

2.1 Otsu's Threshold Method

The Otsu's method [5] is a clustering-based method suggested to identify the threshold that maximizes the distinctiveness of two populations dividing the image. This distinctiveness is expressed by the interclass variance [23]:

$$\sigma_B^2\left(k^*\right)=\max_{1\le k<L}\left(\frac{\left[\mu_T\,\omega(k)-\mu(k)\right]^2}{\omega(k)\left[1-\omega(k)\right]}\right) \tag{II.2}$$

The threshold value is $Th=k^*$.

Where $\omega(k)$ and $\mu(k)$ are the zeroth and first-order cumulative moments of the histogram up to the k^{th} level, respectively. They are defined by: $\omega(k)=\sum_{i=1}^{k}p_i$ and

$\mu(k)=\sum_{i=1}^{k} i.p_i$ and $\mu_T=\sum_{i=1}^{L} i.p_i$ is the total mean level of the original picture, and $L=255$ in case of 8-bit gray scale image. We can summarize this method in these steps:

1. Calculate the histogram of the image.
2. Calculate the mean of the image μ_T .
3. For each k (from 0 to 255):
   - Calculate the zeroth and first-order cumulative moments of the histogram.
   - Calculate the between class-variance σ_B^2 for each value of k .
4. Search k^* that maximizes σ_B^2 , then $Th=k^*$.

2.2 The Iterative Selection

The iterative selection [6] or ISODATA is a clustering-based method which assumes that the distribution of gray-level values is based on a two-class Gaussian mixture model. At each iteration *n*, a new threshold T_n is established using the average of the foreground μ_o and background class means μ_b .

$$T_n=\frac{\mu_o-\mu_b}{2} \tag{II.3}$$

In practice, iterations terminate when the changes $|T_{n2}-T_{n1}|$ become sufficiently small [1], where T_{n2} and T_{n1} are successive iterations. In fact the iterative selection is the iterative version of the minimum error thresholding [24] under certain conditions. The probability of error is given by Eq.(II.4):

$$P_e=\Pi_o.P_m+\Pi_b.P_F \tag{II.4}$$

Where Π_o and Π_b are the prior probabilities of each class: background and object, respectively.

P_m : is the probability of classifying an object pixel as background (the probability of miss)

$$P_m=\int_{-\infty}^{T} P_o dx \tag{II.5}$$

where P_o is the gray level probability distribution of the object.

$$P_o(x) = \frac{1}{\sqrt{2\pi}\sigma_o} \exp\left(-\frac{(x-\mu_o)^2}{2\sigma_o^2} \right) \quad \text{(II.6)}$$

μ_o and σ_o are the mean and the standard deviation of the object distribution respectively.

P_F : is the probability of classifying a background pixel as an object (the probability of false alarm):

$$P_F = \int_T^{+\infty} P_b dx \quad \text{(II.7)}$$

P_b : is the probability distribution of background

$$P_b(x) = \frac{1}{\sqrt{2\pi}\sigma_b} \exp\left(-\frac{(x-\mu_b)^2}{2\sigma_b^2} \right) \quad \text{(II.8)}$$

Where μ_b , σ_b are the mean and the standard deviation of the background distribution.

We can find the threshold value *Th* by minimizing the probability of error given by Eq.(II.4).

$$\frac{dP_e}{dT} = 0 \quad \text{(II.9)}$$

The above expression is minimized when:

$$Th = \left[\begin{array}{l} \dfrac{\sigma_b^2\mu_o - \sigma_o^2\mu_b + \sqrt{-2\sigma_b^2\mu_o\sigma_o^2\mu_b + \sigma_b^2\sigma_o^2\mu_b^2 - 2\sigma_b^4 \ln\left(\frac{\Pi_b.\sigma_o}{\Pi_o\sigma_b}\right)\sigma_o^2 + \sigma_b^2\sigma_o^2\mu_o^2 + 2\sigma_o^4 \ln\left(\frac{\Pi_b.\sigma_o}{\Pi_o\sigma_b}\right)\sigma_b^2}}{\sigma_b^2 - \sigma_o^2}; \\ \\ -\dfrac{-\sigma_b^2\mu_o + \sigma_o^2\mu_b + \sqrt{-2\sigma_b^2\mu_o\sigma_o^2\mu_b + \sigma_b^2\sigma_o^2\mu_b^2 - 2\sigma_b^4 \ln\left(\frac{\Pi_b.\sigma_o}{\Pi_o\sigma_b}\right)\sigma_o^2 + \sigma_b^2\sigma_o^2\mu_o^2 + 2\sigma_o^4 \ln\left(\frac{\Pi_b.\sigma_o}{\Pi_o\sigma_b}\right)\sigma_b^2}}{\sigma_b^2 - \sigma_o^2} \end{array} \right] \quad \text{(II.10)}$$

We have two solutions except when the two classes have the same standard deviation $\sigma_b = \sigma_o = \sigma$, then the threshold T is given by:

$$T_n = \frac{\mu_o - \mu_b}{2} + \frac{\sigma^2}{\mu_b - \mu_o} \ln\left(\frac{\Pi_o}{\Pi_b} \right) \quad \text{(II.11)}$$

If the prior probabilities are equal $\Pi_o = \Pi_b$ then we obtain the equation Eq.(II.3):

$$T_n = \frac{\mu_o - \mu_b}{2} \quad \text{(II.12)}$$

It should be noted that the iterative selection is a special case of minimum error thresholding method with equal prior probabilities and equal variances. In practice we can summarize the iteration selection method in these steps:

1. Calculate the histogram of the image.
2. Given an initial threshold T_{n_1}, the image is divided into two classes; we calculate the mean of each class.
3. Calculate $T_{n_2} = \frac{\mu_o - \mu_b}{2}$.
4. If $|T_{n2} - T_{n1}| < S$ terminate the algorithm. Otherwise $T_{n1} = T_{n2}$ go to step 2.

Where S is a prefixed value that ensures that the difference between T_{n_1} and T_{n_2} is negligible [4].

2.3 Kapur's Entropy Thresholding

Kapur's threshold algorithm [7] is an entropy-based method, which uses the Maximum Entropy (ME) as an optimal criterion for image thresholding. It was first proposed by Pun [25] and later corrected and improved by Kapur *et al*. [7,26]. Basically, the entropy-based thresholding considers an image histogram as a probability distribution, and then selects, as an optimal threshold, the value that yields the ME.

Let p_i be the frequencies of each gray-value in the histogram normalized to the total number of pixels in the image.

$$p_i = \frac{h_i}{N} \tag{II.13}$$

If we assume two classes of pixels, P_f and P_b, where P_f denotes the probability class of foreground pixels, P_b is the probability class of background pixels. The entropies associated with each class are as follows:

$$H_b(\mathrm{T}) = -\sum_{i=0}^{T}\left(\frac{p_i}{P_b}\ln\left(\frac{p_i}{P_b}\right)\right) \tag{II.14}$$

$$H_f(\mathrm{T}) = -\sum_{i=T+1}^{255}\left(\frac{p_i}{P_f}\ln\left(\frac{p_i}{P_f}\right)\right) \tag{II.15}$$

Where: $P_b = \sum_{i=0}^{T} p_i$, $P_f = \sum_{i=T+1}^{255} p_i$ and $P_f + P_b = \sum_{i=0}^{255} p_i = 1$

The information between the two classes H is defined as:

$$H(\mathrm{T}) = H_b(\mathrm{T}) + H_f(\mathrm{T}) \tag{II.16}$$

The gray-value T that maximize $H(\mathrm{T})$ is the threshold value. We can summarize this method in these steps:

1. Calculate the histogram of the image.
2. For each T (from 0 to 255):
   - Calculate $H_b(\mathrm{T})$ and $H_f(\mathrm{T})$.
   - Calculate H for each value of T.
3. Search T that maximizes H , then $Th = T$.

2.4 Ramesh's Threshold Method

Ramesh's threshold algorithm [8] is a histogram shape method, based on minimizing the sum of variances of gray levels of the background and foreground pixels. The solution for Th is obtained by iterative search.

$$R(Th) = \min_{1 \le T < 255} \left[\sum_{i=0}^{T} \left[m_1(T) - i \right]^2 + \sum_{i=T+1}^{L} \left[m_2(T) - i \right]^2 \right] \tag{II.17}$$

Where $m_1(T) = \dfrac{\sum_{i=0}^{T} i \,.\, p_i}{\sum_{i=0}^{T} p_i}$ and $m_2(T) = \dfrac{\sum_{i=T+1}^{L} i \,.\, p_i}{\sum_{i=T+1}^{L} p_i}$

We can sum up this method in the following steps:

1. Calculate the histogram of the image.
2. For each T (from 0 to 255):
   - Calculate m_1 and m_2.
   - Calculate the sum of variances R for each value of T.

Search T that minimizes R .

2.5 Moment Preserving Threshold (Tsai's Threshold)

It is an attribute-based method [9]; it considers the gray-level image as the blurred version of an ideal binary image. The thresholding is established so that the first three gray-level moments match the first three moments of the binary image [1].

$$m_k = m'_k \text{ , k=1,2,3} \tag{II.18}$$

The gray-level moments m_k and binary image moments m'_k are defined for bi-level segmentation, respectively, as:

$$m_k = \sum_{i=0}^{L} p_i . i^k \tag{II.19}$$

$$m'_k = P_b m_b^k + P_f m_f^k \tag{II.20}$$

Then the equations for the first three moments are

$$\begin{cases} m_1 = P_b m_b^1 + P_f m_f^1 \\ m_2 = P_b m_b^2 + P_f m_f^2 \\ m_3 = P_b m_b^3 + P_f m_f^3 \end{cases} \tag{II.21}$$

In order to find the optimal threshold Th, by solving Eq.(II.21), P_b is first found to be [4],[27]:

$$P_b = \frac{G - m_1}{\sqrt{c_1^2 - 4c_0}} \tag{II.22}$$

In which,

$$c_0 = \frac{m_1 m_3 - m_2^2}{m_2 - m_1^2}, \quad c_1 = \frac{m_1 m_2 - m_3}{m_2 - m_1^2}, \quad G = \frac{1}{2}\left[\sqrt{c_1^2 - 4c_0} - c_1\right]$$

The threshold value Th is obtained from the cumulative distribution of the gray-level histogram of the image by choosing Th as the P_b-tile, which satisfies:

$$P_b = \sum_{i \le T} p_i \tag{II.23}$$

2.6 GMM threshold method

A Gaussian mixture model (GMM) is a category of probabilistic model which states that all generated data points are derived from a mixture of finite Gaussian distributions that has no

known parameters. The parameters for Gaussian mixture models are derived either from maximum a posteriori estimation or an iterative expectation maximization algorithm from a prior model which is well trained. Gaussian mixture models are very useful when it comes to modeling data and classification, especially data which comes from several groups. In our case (thresholding), the Expectation Maximization (EM) algorithm is developed to estimate the number of Gaussian mixture of such histograms and their corresponding parameterization for each pixel of the image $I(x,y)$ [10]. The algorithm below shows the steps to compute the optimal threshold.

1. Given an image with gray-level range, compute the gray-level frequency distribution and histogram.
2. Initiate Gaussian components μ, σ .
3. The EM is used to determine the Gaussian parameters by maximizing this Eq:

$$P(x,\psi) = \sum_{i=1}^{k} w_i \; f\,(x, \mu_i, \sigma_i) \tag{II.24}$$

where μ is the mean and σ is the standard deviation, w_i is the weighted associated with each Gaussian, ψ is the parameters of the Gaussian $\psi(\mu,\sigma,w)$, f is the Gaussian probability density function given by:

$$f\,(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(\frac{(x-\mu)^2}{2\sigma^2} \right) \tag{II.25}$$

4. If the estimation error for step (3) is less than a predefined threshold or the number of iteration is greater than D , stop and output results. Otherwise, go to step (3).
5. Add a new Gaussian component and perform step (2) again.
6. In the end, we choose the optimal threshold which is the average of these means μ_m.

$$T_{opt} = \frac{1}{m} \sum_{i=1}^{m} \mu_m \tag{II.26}$$

3. DIFFERENTIAL MOTION DETECTION ALGORITHMS

The movement in the video is an information source which has produced an important scientific activity in several fields such as video surveillance. Motion detection has been a key component in video surveillance systems by detecting intrusions and triggering alarms in different situations and domains such as commercial (airports, shops malls and banks), industrial (factories and plants) and countries security (border surveillance). More recently,

government agencies, schools and even individuals are turning toward these systems as a means to increase security [28]. In this section, a review of basic motion detection algorithms which are used in this comparison study is presented.

3.1 Frame Difference (Delta frame)

The frame difference algorithm is a simple method to extract moving objects. It consists of calculating the absolute difference between the previous image I_{t-1} and the current image I_t, Eq.(II.27).

$$\zeta(x,y)=\left|\frac{dI(x,y)}{dt}\right|=\left|\Delta I_{t,t-1}(x,y)\right|=\left|I_t(x,y)-I_{t-1}(x,y)\right| \tag{II.27}$$

3.2 Running Average Filter

Running Average filter (RAF) or recursive background subtraction method is a method in which the background is constructed by calculating the mean value of the previous N frames, in order to update the first background image taking into account new static objects in the scene. A background image is obtained as follows [29]:

$$B(x,y)=\frac{1}{t}\sum_{\tau=1}^{t} I(x,y,\tau) \tag{II.28}$$

Eq.(II.28) can be also computed recursively:

$$B(x,y,t)=\frac{t-1}{t}B(x,y,t-1)+\frac{1}{t}\mathrm{I}(x,y,t-1) \tag{II.29}$$

However, this approach may fail when the lighting conditions change significantly over time. It is natural to assume that most recent images contribute more to the background than the previous image. This can be achieved by replacing l/t in Eq.(II.28) by a constant α so that each image contribution to the background decreases exponentially as it recedes into the past [18]. The background obtained by an exponential forgetting method is given by:

$$B(x,y,t)=(1-\alpha)B(x,y,t-1)+\alpha\mathrm{I}(x,y,t-1) \tag{II.30}$$

where $\alpha\in[0,1]$ is a time constant that specifies how fast new information supplants old observations, we have taken $\alpha=0.05$ in all experiments. Using this background image, we perform the difference as in Eq.(II.31).

$$\zeta_2(x,y)=\left|B(x,y)-I_t(x,y)\right| \tag{II.31}$$

3.3 Hybrid Motion Detection Method

The hybrid motion detection algorithm consists of combining the recursive background subtraction method with the three frame difference method.

The three frame difference gives a robust detection for sufficient large movements (large objects, objects near camera), and deletes noises.

More than that, it adapts to changes in luminance. The three frame difference is given by Eq.(II.32), (II.33) and (II.34):

$$\zeta'(x,y)=\left|\Delta I_{t,t-1}(x,y)\right|=\left|I_t(x,y)-I_{t-1}(x,y)\right| \quad \text{(II.32)}$$

$$\zeta''(x,y)=\left|\Delta I_{t,t+1}(x,y)\right|=\left|I_t(x,y)-I_{t+1}(x,y)\right| \quad \text{(II.33)}$$

$$\zeta_1(x,y)=\text{Min}\left(\zeta'(x,y),\zeta''(x,y)\right) \quad \text{(II.34)}$$

Where $\zeta'(x,y)$ is the frame difference between the frames *t* and *t-1*. $\zeta''(x,y)$ is the frame difference between the frames *t* and *t+1* and $\zeta_1(x,y)$ is the three frame difference.

However, the main problem of this method is the incomplete segmentation and the holes left in the detected object.

Based on this result, the hybrid method takes the maximum of the three frame difference and the recursive background subtraction, in order to fill up the holes left by the first method (Fig.II.1).

$$\zeta(x,y)=Max\left(\zeta_1(x,y),\zeta_2(x,y)\right) \quad \text{(II.35)}$$

All these motion detection methods are followed by a thresholding step as described in Eq.(II.1) to detect the region of changes. In the hybrid motion detection method, we will need to threshold each result issued from the three frame difference and adaptive background subtraction methods (Fig.II.1).

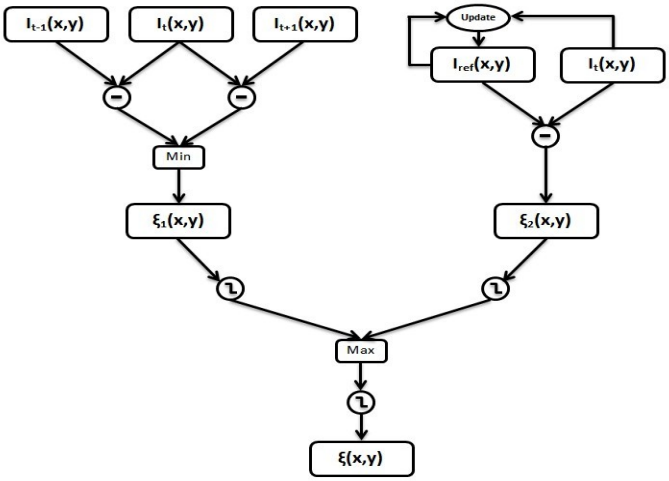


Fig.II.1. Hybrid motion detection diagram

4. EXPERIMENTAL RESULTS, EVALUATION AND DISCUSSION

We have applied all threshold methods described in Sec.2 on motion detection algorithms of Sec.3. Fig.II.2 shows the background images of four data used for performance evaluation. We have chosen different scenes according to illumination condition (Indoor/Outdoor area) and different objects in background (trees, flags, static objects). The shortest video lasts 5 minutes with a frame rate of 25 fps, the resolution of the (a,b,c) scenes is 680x480 while the (d) scene has a resolution of 720x576.

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

Fig.II.2. Experimental environment background scenes

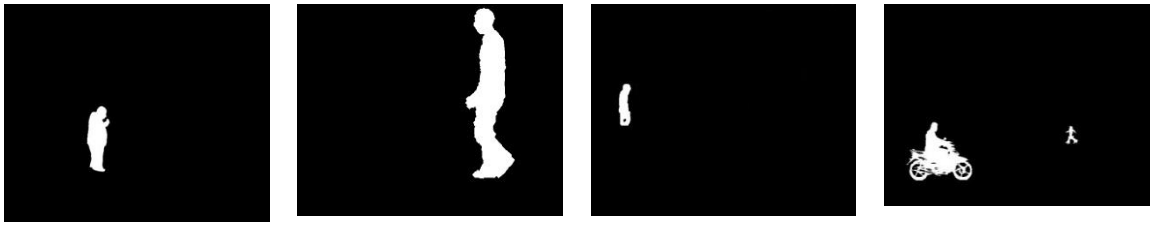
Fig.II.3 shows the experimental environment scenes with objects in motion tested with all the different methods mentioned in sections 2 and 3. The objects in motion are with different size and they are in different distance from the camera. These images have been used to develop the ground truth for pixel-based evaluation.

(a) 283th frame (b) 73th frame (c) 105th frame (d) 6060th frame

Fig.II.3. Experimental environment scenes with objects in motion

4.1 Ground Truth Generation

The Ground truth has been generated through many processes. We have applied at first the sobel edge operator on the image resulting from background subtraction method, and then we performed an area opening to remove any connected component that has an area less than P pixels. After that, we have smoothed the results using twice the 'erode' morphological operator with a diamond structuring element. We have avoided using any threshold methods in the generation of the ground truth to not affect the results of evaluation. Finally, the ground truth image result was treated manually to approximate the real results. We should note that the manual operation does not deliver the same results and differ considerably from one human observer to another, even when they are provided with a common set of guidelines. The same human observer can even generate different segmentations for the same data at two different times [22]. The ground truth images generated from these scenes are presented in Fig.II.4.



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

Fig.II.4. Ground truth images

4.2 Pixel-based evaluation

The pixel-based evaluation is based on four quantities defined in [21,22] as:

- True positives (TP): the number of foreground pixels correctly detected.
- False positives (FP): the number of background pixels incorrectly detected as foreground (also known as false alarms).
- True negatives (TN): the number of background pixels correctly detected.
- False negatives (FN): the number of foreground pixels incorrectly detected as background (also known as misses).

Based on these four quantities, we can define these metrics:

- Percentage of Correct Classification (PCC): this measure is the most common. It is defined as the pixels correctly classified and given by the following equation:

$$PCC = \frac{TP + TN}{TP + TN + FN + FP} \tag{II.36}$$

This metric however can be misleading when class distribution is unbalanced, e.g. when the data set contains fewer foreground object pixels and larger percentage of background pixels [30]. To rectify this issue, the Jaccard and the Yule coefficients have been introduced.

- The Jaccard Coefficient is defined by:

$$JC = \frac{TP}{TP + FP + FN} \tag{II.37}$$

- The Yule Coefficient is defined by:

$$YC = \left| \frac{TP}{TP + FP} + \frac{TN}{TN + FN} - 1 \right| \tag{II.38}$$

From the definitions of these metrics, we can notice that the larger values lead to the best performance, by promoting the correct detected pixels. However, as mentioned above, the PCC metric gives a misleading estimate when the amount of change is small compared to the overall image, this is exactly our case, a very high rating can be achieved even if the segmentation is not good. To overcome this problem an average of these three coefficients is calculated to define the best segmentation Eq.(II.39).

$$Av.Coef = \frac{PCC/100 + JC + YC}{3} \tag{II.39}$$

4.3 Results and Discussion

In this section, we present the image results of applying different threshold methods on motion detection algorithms, followed by a pixel-based evaluation. A general comparison between the best results is discussed.

We start with the first motion detection algorithm presented in Sec.3.1, the frame difference method; Fig.II.5 presents the results of this method.

From these results, we note that the Kapur's threshold method gives the best result, in indoor and outdoor areas, by extracting entirely the shape of the moving objects. The second best method is the GMM threshold method which gives good results in indoor and outdoor scenes. It is followed by Otsu's and the iterative selection methods which give almost the same results. Then, the fourth one which is the Tsai's method gives a good result in indoor area but fails to extract the moving objects in outdoor areas especially the ones which are far

from the camera. Ramesh's threshold method fails to extract the shape of the moving object considering many pixels as background.

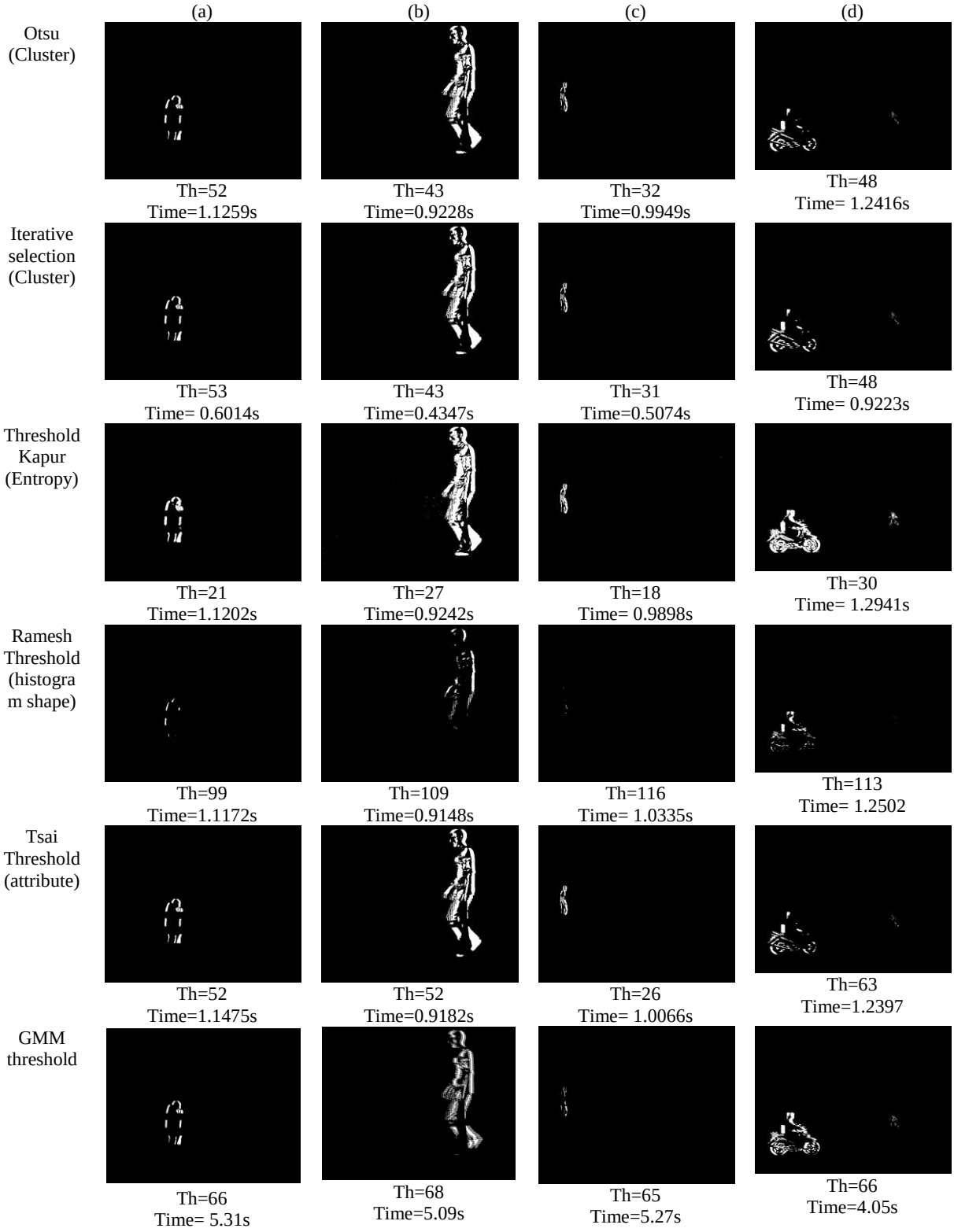


Fig.II.5. Thresholding results and time execution applied on frame difference method

Table.II.1 presents the pixel-based evaluation of these methods applied on frame difference method in indoor and outdoor area. We note that we have high rating for PCC. This is due to the small area of moving object compared to the whole image. However, the Jaccard and the Yule coefficients overcome this problem, the mean of this coefficient is given to pick up easily the difference.

For time of execution, we see that the iterative selection is the fastest one (Fig.II.5). The execution time of the other methods are approximately the same, except for the GMM which has long time of execution due to computational load.

Table II.1 Pixel-based evaluation of different threshold methods applied on frame difference method. larger values indicate better performance

		Scene 1 (Indoor)	Scene 2 (Indoor)	Scene 3 (outdoor)	Scene 4 (outdoor)	Average Coef. Scene1(indoor)	Average Coef. Scene2(indoor)	Average Coef. Scene3(outdoor)	Average Coef. Scene4 (outdoor)
Otsu	PCC (%)	98.26	95.21	99.55	97,06	0.5026	0.7095	0.7627	0.5952
	JC	0.1009	0.4109	0.3186	0.2706				
	YC	0.4245	0.7657	0.9742	0.5444				
IS	PCC (%)	98.27	95.21	99.55	97,06	0.5024	0.7095	0.7664	0.5952
	JC	0.1000	0.4109	0.3288	0.2706				
	YC	0.4247	0.7657	0.9749	0.5444				
Kapur	**PCC (%)**	**98.27**	**95.86**	**99.63**	**97.14**	**0.5267**	**0.7441**	**0.8037**	**0.6182**
	JC	**0.1493**	**0.5100**	**0.4537**	**0.3368**				
	YC	**0.4483**	**0.7637**	**0.9611**	**0.5464**				
Ramesh	PCC (%)	98.25	93.12	99.35	96.84	0.4519	0.5831	0.6673	0.5154
	JC	0.0424	0.0813	0.0151	0.0753				
	YC	0.3310	0.7370	0.9935	0.5025				
Tsai	PCC (%)	98.26	94.93	99.59	96,98	0.5026	0.6952	0.7827	0.5733
	JC	0.1009	0.3665	0.3805	0.2154				
	YC	0.4245	0.7700	0.9717	0.5348				
GMM	PCC (%)	98.30	95.22	99.67	97.03	0.5387	0.7050	0.8028	0.6083
	JC	0.1614	0.4355	0.5427	0.3302				
	YC	0.4717	0.7274	0.8690	0.5244				

Fig.II.6 presents the results of applying the threshold methods on the recursive background subtraction method (running average filter). These results show that Otsu's threshold and the iterative selection methods give the best results in indoor and outdoor scenes. The GMM also give a good result. The method succeeds on eliminating the shadow in indoor scenes (a & b) and in outdoor scenes the slow movement of the tree branch are also eliminated.

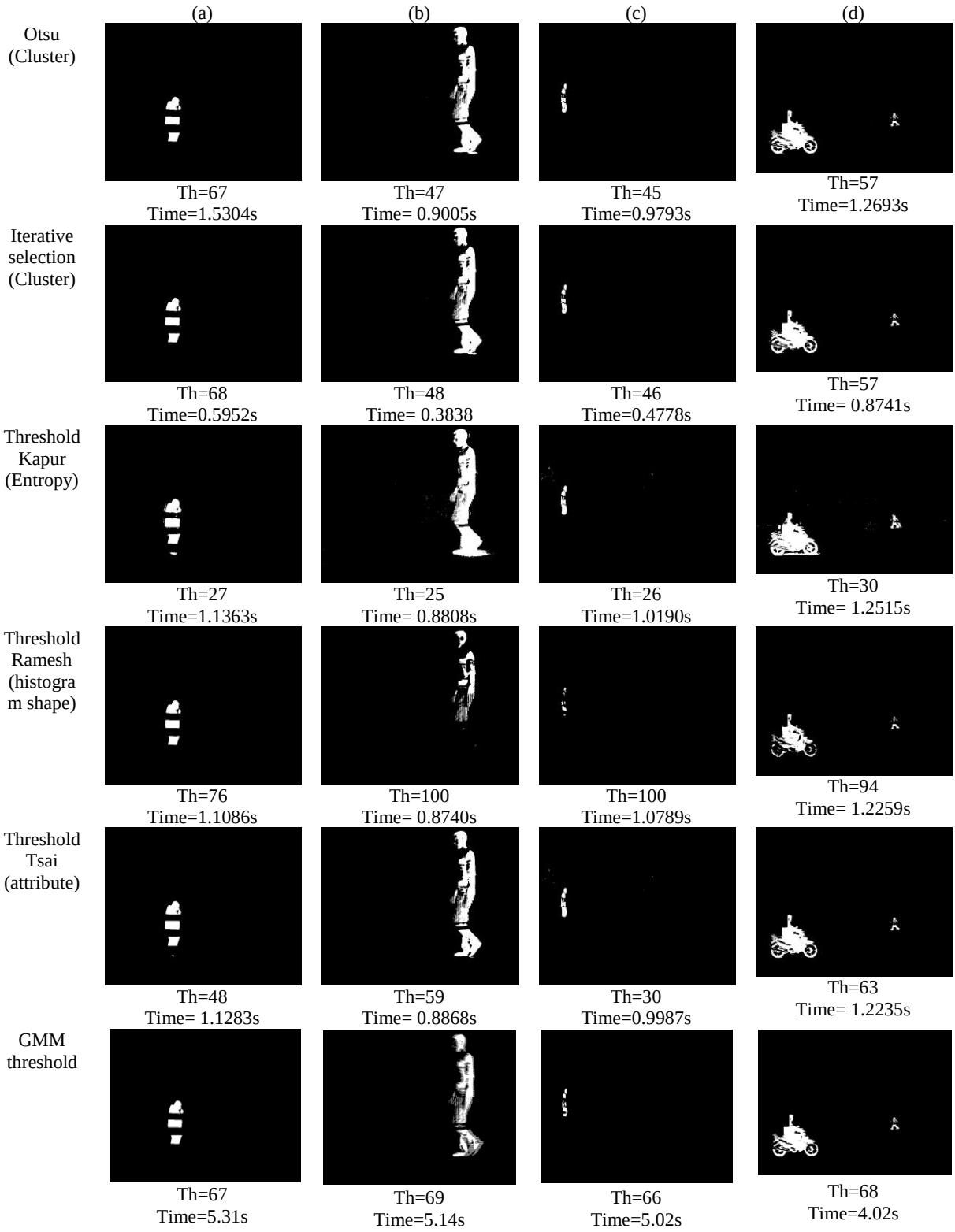


Fig.II.6. Thresholding results and time execution applied on running average filter method

Unlike the previous test, Kapur's threshold method fails in all the scenes. In indoor area, cases of (a) and (b), the shadow of moving persons appears which present a great problem impairing feature extraction of objects in motion. For the outdoor scenes (c) and (d), the small movements of tree branches and leaves appear creating false detections. This is also the case

for the Tsai's threshold method for outdoor scenes, but in indoor scenes results are better. For Ramesh's threshold method, results are good but the shapes of the moving objects are not well detected.

The execution time has increased due to the recursive steps in this method. We can see that the iterative selection is the fastest method. Table.II.2 presents the pixel-based evaluation of this method. We notice also, that the Jaccard and Yule coefficients have been increased considerably compared to the delta frame method. This is due to the holes filled by this background detection method unlike the delta frame.

Table II.2 Pixel-based evaluation of different threshold methods applied on running average filter. Larger values indicate better performance

		Scene 1 (Indoor)	Scene 2 (Indoor)	Scene 3 (outdoor)	Scene 4 (outdoor)	Average Coef. Scene1(indoor)	Average Coef. Scene2(indoor)	Average Coef. Scene3(outdoor)	Average Coef. Scene4 (outdoor)
Otsu	**PCC (%)**	**99.10**	**98.12**	**99.70**	**98.17**	**0.7860**	**0.8922**	**0.8351**	**0.7484**
	JC	**0.5064**	**0.7517**	**0.5633**	**0.5436**				
	YC	**0.8606**	**0.9439**	**0.9451**	**0.7199**				
IS	**PCC (%)**	**99.10**	**98.11**	**99.70**	**98.17**	**0.7860**	**0.8926**	**0.8337**	**0.7484**
	JC	**0.5056**	**0.7508**	**0.5594**	**0.5436**				
	YC	**0.8615**	**0.9461**	**0.9447**	**0.7199**				
Kapur	PCC (%)	99.09	98.03	99.62	97.99	0.7723	0.8657	0.7570	0.7272
	JC	0.5368	0.7653	0.5295	0.5541				
	YC	0.7893	0.8517	0.7454	0.6478				
Ramesh	PCC (%)	99.09	95.20	99.44	98.00	0.7856	0.7470	0.7131	0.7334
	JC	0.4987	0.3482	0.1505	0.4514				
	YC	0.8672	0.9409	0.9944	0.7688				
Tsai	PCC (%)	99.11	97.80	99.66	98.15	0.7857	0.8798	0.7850	0.7466
	JC	0.5225	0.7053	0.5456	0.5315				
	YC	0.8435	0.9561	0.8129	0.7268				
GMM	PCC (%)	99.12	97.08	99.66	98.07	0.7803	0.8291	0.8172	0.7354
	JC	0.5436	0.6256	0.5022	0.5569				
	YC	0.8061	0.8910	0.9530	0.6688				

Fig.II.7 presents results of applying threshold methods on hybrid background detection. In this case, we have three threshold values, a threshold for the recursive background detection method, two thresholds for the two delta frame methods. Results show that Otsu, iterative selection and GMM are the best threshold methods deleting isolated points and preserving the

shape of the objects in motion. In second rank, we found the Ramesh threshold method where the shape of the moving object is well extracted. However, the appearance of the tails caused by the wrong selected value of α alters the results.

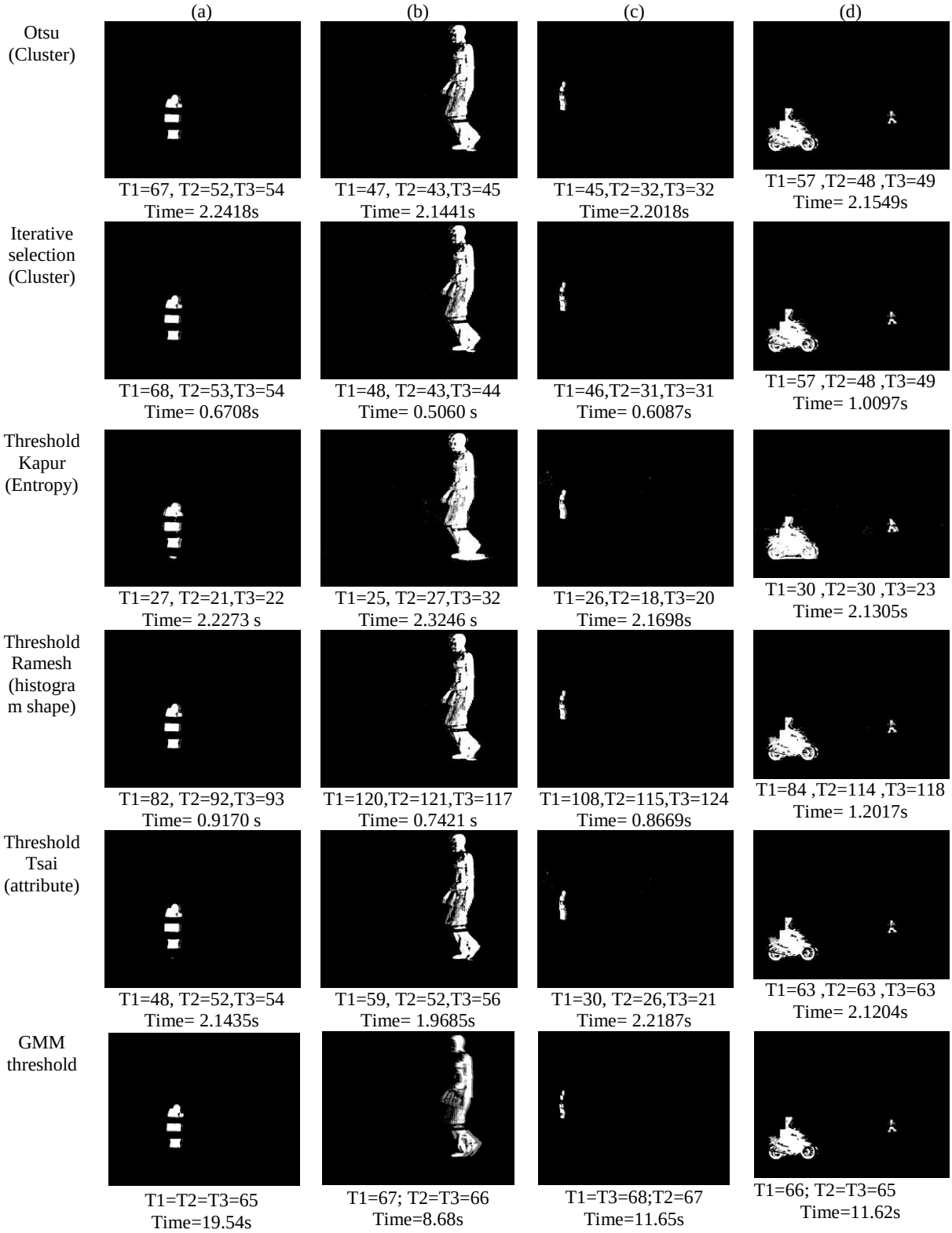


Fig.II.7. Thresholding results applied on hybrid motion detection method

Tsai's threshold method gives good results only in indoor area. In outdoor area, the presence of unwanted movement of tree leaves, in scene (c), gives wrong results and decreases the average score of evaluation coefficients (Table II.3). Kapur's method does not succeed in giving the correct result in this case. The iterative selection has the advantage of being the fastest method.

From these comparisons, we note that the Otsu's threshold method and the iterative selection method give the best results in all cases. Actually, these two methods lead to similar results, under the same condition of standard deviation. A detailed proof is presented by Xiangyang et al. in [31]. We note also that the GMM method has approximately the same value of threshold. This is due to the distribution of the pixel intensities of the foreground image which cannot be modeled ideally by a mixture of Gaussians (a parametric model, see Fig.II.8).

It should be noted that the background subtraction method presents the drawback of considering an object which enters the scene and stops (it becomes static) as a moving object. For that reason we have used the recursive background detection algorithm to overcome this problem.

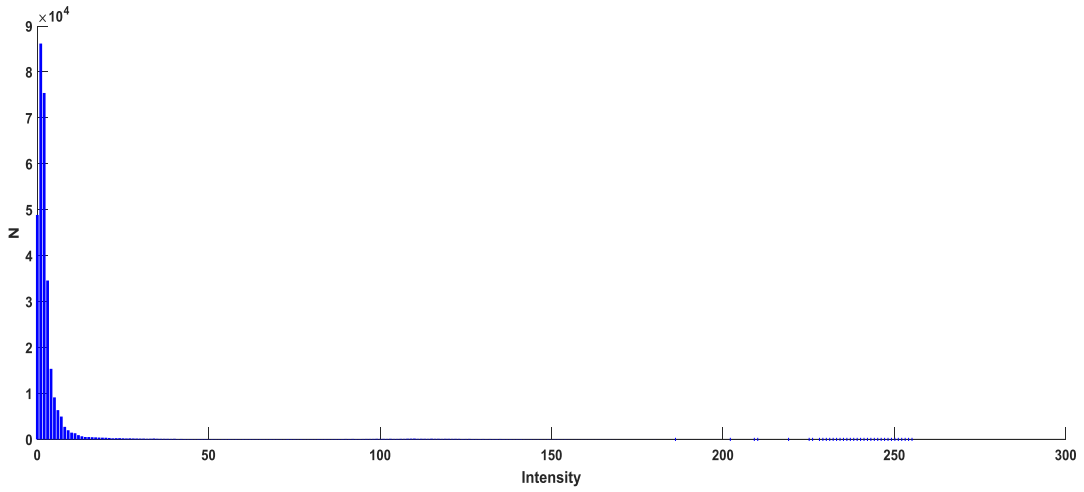


Fig.II.8. Histogram of the running average method applied to scene 4 before thresholding

Table II.3 Pixel-based evaluation of different threshold methods applied on hybrid background detection. Larger values indicate better performance

		Scene 1 (Indoor)	Scene 2 (Indoor)	Scene 3 (outdoor)	Scene 4 (outdoor)	Average Coef. Scene1(indoor)	Average Coef. Scene2(indoor)	Average Coef. Scene3(outdoor)	Average Coef. Scene4 (outdoor)
Otsu	**PCC (%)**	**99.10**	**97.71**	**99.71**	**98.49**	**0.7797**	**0.8466**	**0.8272**	**0.7837**
	JC	**0.5131**	**0.7307**	**0.6041**	**0.6253**				
	YC	**0.8350**	**0.8322**	**0.8805**	**0.7409**				
IS	**PCC (%)**	**99.10**	**97.67**	**99.71**	**98.49**	**0.7800**	**0.8442**	**0.8263**	**0.7837**
	JC	**0.5126**	**0.7265**	**0.6017**	**0.6253**				
	YC	**0.8363**	**0.8296**	**0.8801**	**0.7409**				
Kapur	PCC (%)	99.07	97.12	99.62	98.30	0.7603	0.8042	0.7522	0.7607
	JC	0.5394	0.7011	0.5644	0.6328				
	YC	0.7510	0.7404	0.6962	0.6664				
Ramesh	PCC (%)	99.10	97.69	99.71	98.48	0.7796	0.8446	0.8283	0.7833
	JC	0.5139	0.7290	0.6068	0.6264				
	YC	0.8341	0.8281	0.8810	0.7388				
Tsai	PCC (%)	99.11	97.60	99.66	98.39	0.7799	0.8470	0.7821	0.7745
	JC	0.5273	0.7057	0.5835	0.5930				
	YC	0.8213	0.8594	0.7662	0.7468				
GMM	PCC (%)	99.68	98.16	97.13	99.10	0.8274	0.7463	0.8180	0.7705
	JC	0.5313	0.5930	0.6546	0.5509				
	YC	0.9543	0.6644	0.8283	0.7697				

5. CONCLUSION

In this chapter, we have presented an objective and a subjective evaluation and comparison of six threshold methods applied on three differential motion detection algorithms. Results and a discussion were drawn from the application of these different combinations on four real scenes. The results showed that the Otsu's and the iterative selection methods give the best results in all cases, followed by the GMM threshold method. The Kapur's threshold method gives good results just with the delta frame method, but with the other methods, the presence of shadows impair the segmentation. Unlike Kapur's threshold method, the Tsai's and Ramesh's threshold method have an acceptable performance on background subtraction techniques with a little advantage to Tsai's method. But with the delta frame method, these methods do not succeed on extracting the shape of the moving objects. The objective evaluation using the average of the three coefficients give similar results to the observed

images. However, we believe that more measures and evaluation metrics, the use of other threshold methods and other motion detection algorithms, are necessary to expand this work.

For that, the next chapter, will discuss the state of the art of all motion detection techniques, Moreover, a benchmark of twelve motion detection methods applied on large datasets that includes many challenge has been studied in order to let us choose the appropriate method for our application in means of accuracy and execution time.

REFERENCES

[1] Sezgin M, Sankur B, "Survey over Image Thresholding Techniques and Quantitative Performance Evaluation". Journal of Electronic Imaging, 13: 146-165, (2004).

[2] S.U. Le, S.Y. Chung, R.H. Park, "A Comparative Performance Study of Several Global Thresholding Techniques for Segmentation", Computer Vision, Graphical Models and Image Processing, 52 171-190, (1990).

[3] H.B. Mitchell, Image Fusion: Theories, Techniques and Application, Springer Science & Business Media, (2010).

[4] Anna Fabijanska, "A survey of thresholding algorithms on yarn images" ,in MEMS-TECH 2010, Polyana-Svalyava ,Ukraine , 20–23 April (2010).

[5] N. Otsu, ''A threshold selection method from gray level histograms,'' IEEE Trans. Syst. Man Cybern. SMC-9, 62–66 (1979).

[6] T. W. Ridler and S. Calvard," Picture Thresholding Using an Iterative Selection Method", IEEE Transactions On Systems, Man, And Cybernetics, Vol. sMC-8, NO. 8, AUGUST (1978).

[7] J. N. Kapur, P. K. Sahoo, and A. K. C. Wong, "A new method for gray-level picture thresholding using the entropy of the histogram", Graph. Models Image Process. 29, 273–285, (1985).

[8] N. Ramesh, J. H. Yoo, and I. K. Sethi, "Thresholding based on histogram approximation", IEEE Proc. Vision Image Signal Process. 142(5), 271–279, (1995).

[9] W. H. Tsai, ''Moment-preserving thresholding: A new approach,'' Graph. Models Image Process. 19, 377–393,(1985).

[10] Z.-K. Huang and K.-W. Chau, "A new image thresholding method based on gaussian mixture model," Applied Mathematics and Computation, vol. 205, no. 2, pp. 899–907,2008.

[11] R.T. Collins et al., A system for video surveillance and monitoring: VSAM final report, CMU-RI-TR-00-12, Technical Report, Carnegie Mellon University, (2000).

[12] K. Sehairi, C. Benbouchama, and F. Chouireb, "Real Time Implementation on FPGA of Moving Objects Detection and Classification," International Journal of Circuits, Systems and Signal Processing, vol.9, pp 160-167, 2015.

[13] Xiaoshi Zheng; Yanling Zhao; Na Li; Huimin Wu, "An Automatic Moving Object Detection Algorithm for Video Surveillance Applications," International Conference on Embedded Software and Systems, 2009. ICESS '09., vol., no., pp.541,543, 25-27 May (2009).

[14] Widyawan; Zul, M.I.; Nugroho, L.E., "Adaptive motion detection algorithm using frame differences and dynamic template matching method," 9th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI), vol., no., pp.236,239, 26-28 Nov. (2012).

[15] Liang Wang, Weiming Hu, Tieniu Tan," Recent developments in human motion analysis", Journal Pattern Recognition, Vol. 36, No. 3. pp. 585-601, March (2003).

[16] 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, Vol. 34, No. 3, August (2004).

[17] Jorge Hiraiwa, Enrique Vargas, Sergio Toral," An FPGA based Embedded Vision System for Real-Time Motion Segmentation", IWSSIP 2010 - 17th International Conference on Systems, Signals and Image Processing, Rio de Janeiro, Brazil,17-19 Jun, (2010).

[18] Xiao-jun Tan, Jun Li and Chunlu Liu," A video-based real-time vehicle detection method by classified background learning", World Transactions on Engineering and Technology Education, Vol.6, No.1, (2007).

[19] Ruolin Zhang, Jian Ding," Object Tracking and Detecting Based on Adaptive Background Subtraction", International Workshop on Information and Electronics Engineering, Vol 29, , p 1351–1355,(2012).

[20] Yan Zhao; Jiao-Min Liu, "An improved method for human motion detection and application," 3rd International Congress on Image and Signal Processing (CISP), vol.1, no., pp.258,260, 16-18 Oct. (2010).

[21] Rosin P, Ioannidis E. Evaluation of global image thresholding for change detection. Patt. Recog. Letters, 24(14): 2345-2356; October (2003).

[22] S. Elhabian, K. El-Sayed, S. Ahmed, "Moving Object Detection in Spatial Domain using Background Removal Techniques - State-of-Art", Recent Patents on Computer Science, Volume 1, Number 1, pages 32-54, January (2008).

[23] Maria Petrou and Costas Petrou, Image Processing: The Fundamentals, Second Edition, Wiley, (2010).

[24] Jing-Hao Xue, Yu-Jin Zhang, "Ridler and Calvard's, Kittler and Illingworth's and Otsu's methods for image thresholding", Pattern Recognition Letters, 33 ,793–797, (2012).

[25] T. Pun, ''A new method for gray-level picture threshold using the entropy of the histogram,'' Signal Process. 2(3), 223–237 ,(1980).

[26] C.-I Chang, Y. Du, J. Wang, S.-M. Guo and P.D. Thouin," Survey and comparative analysis of entropy and relative entropy thresholding techniques", IEE Proceedings - Vision, Image and Signal Processing, Volume 153, Issue 6, p. 837 – 850, December (2006).

[27] Shitu Luo; Qi Zhang; Feilu Luo; Yanling Wang; Zhiyong Chen, "An improved moment-preserving auto threshold image segmentation algorithm," International Conference on Information Acquisition, 2004. Proceedings, vol., no., pp.316,318, 21-25 June, (2004).

[28] L. Ovseník, A. Kažimírová Kolesárová, J. Turán, "Object detection in video surveillance systems", Journal of Electronic and Computer Engineering 3, 137-143, (2010).

[29] Piccardi, M., "Background subtraction techniques: a review" IEEE International Conference on Systems, Man and Cybernetics, vol.4, no., pp.3099,3104 vol.4, 10-13 Oct. (2004).

[30] Nicolas Lomnie, Daniel Racoceanu, and Alexandre Gouaillard, Advances in Bio-Imaging: From Physics to Signal Understanding Issues State-Of-The-Art and Challenges, chp.13, pp,215, Springer Publishing Company, Incorporated, (2012).

[31] Xiangyang Xu, Shengzhou Xu, Lianghai Jin, Enmin Song, "Characteristic analysis of Otsu threshold and its applications", 32, Pattern Recognition Letters, 956–961, (2011).

Chapter III

Comparative study of motion detection methods for video surveillance systems

No one model that works best for every problem

“No Free Lunch” theorem

1. INTRODUCTION

Motion detection, which is the fundamental step in video surveillance, aims to detect regions corresponding to moving objects. The resultant information often forms the basis for higher-level operations that require well-segmented results, such as object classification and action or activity recognition. However, motion detection suffers from problems caused by source noise, complex backgrounds, variations in scene illumination, and the shadows of static and moving objects.

Various methods have been proposed to overcome these problems by retaining only the moving object of interest. These methods are classified [1-3] into three major categories: background subtraction, [4,5] temporal differencing, [6,7] and optical flow [8,9]. Temporal differencing is highly adaptive to dynamic environments. However, it exhibits generally poor performance in extracting all relevant feature pixels. Therefore, techniques such as morphological operations and hole filling are applied to effectively extract the shape of a moving object. Background subtraction provides the most complete feature data, but it is extremely sensitive to dynamic scene changes due to lighting and extraneous events. Several background modelling methods have been proposed to overcome these problems. Bouwmans [10] classified these advanced methods into seven categories: basic background modelling [11,12,13] , statistical background modelling [10,14-16] , fuzzy background modelling [17,18], background clustering [19,20], neural network background modelling [21-24], wavelet background modelling [25,26] and background estimation [27,28,29]. Furthermore, Goyette *et al.* [30] categorized background modelling techniques into six families: basic [31-34], parametric [14,15,19,35-37] , non-parametric and data-driven [16,38-41] , matrix decomposition [42-45], motion segmentation [46-48] and machine learning [21,22,49-52]. Sobral and Vacavant [12] adopted a similar categorization for motion detection methods: basic (frame difference, mean and variance over time) [18,54-57], statistical [14,15,58-60], fuzzy [17,18,61-63], neural and neuro-fuzzy [21,22,64,65] and other models (PCA [42],Vumeter [66]). Optical flow can be used to detect independently moving objects even in the presence of camera motion. However, most optical flow methods are computationally complex and cannot be applied to full-frame video streams in real time without specialized hardware [13]. Recent categories have emerged in the last few years such as: advanced non parametric modelling (Vibe [39], PBAS [40], Vibe+ [68]),

background modelling by decomposition into matrices [69-74] and background modelling by decomposition into tensors [75-77].

The remainder of this chapter is organized as follows. Section 2 presents the different surveys and comparative studies conducted on motion detection techniques, and discusses the different categorization of these techniques and the different datasets used for evaluation. Section 3 reviews the motion detection algorithms used in this study (frame difference techniques, background modelling techniques and energy-based methods). Section 4 provides a detailed explanation of the evaluation metrics used to score and evaluate the above-mentioned methods. Section 5 presents and discusses the results for different categories. Finally, Section 6 concludes the chapter.

2. RELATED WORKS

2.1 Survey papers

Several surveys on motion detection methods have been presented in the last decade. The authors tried to detail the algorithms used, categorize them and explain their different steps such as post-processing, initialization, background modelling and foreground generation. Bouwmans [78] presented the most complete survey of traditional and recent methods for foreground detection with more than 300 references. These methods were categorized by the mathematical approach used. In addition, the author explored the different datasets available, existing background subtraction libraries and codes. Elhabian *et al.* [79] provided a detailed explanation of different background modelling methods. Specifically, they explained how models can be updated and initialized, and explored ways to measure and evaluate their performance. Morris *et al.*[80] reviewed three pixel-wise subtraction techniques and compared their capabilities in seven points: resource utilization, computational speed, robustness to noise, precision of the output (no numerical results were provided), complexity of the environment, level of autonomy and scalability. Bouwmans [81] in another work presented a review paper in which he classifies the different improvement techniques made to principal component analysis method (Eigen Backgrounds) and compared these techniques with Gaussian detection methods and kernel density estimation using Wallflower datasets. Cuevas *et al.* [82] presented a state of the art of different techniques for detecting stationary foreground objects e.g. abandoned luggage, people remaining temporarily static or objects removed from the background. The authors addressed the

main challenges in this field with different datasets available for testing these methods. Bux *et.al* [83] gave a complete survey on human activity recognition (HAR), providing the state of the art of foreground segmentation, feature extraction and activity recognition, which constitute the different phases of HAR systems. In foreground segmentation phase, the authors first categorized all methods to background construction-based segmentation for static cameras and foreground extraction-based segmentation for moving cameras. They detailed the different steps of background construction (Initialization, maintenance and foreground detection) and classified these methods into five models: basic, statistical, fuzzy, neural network and others. Cristani *et al.* [84] presented a comprehensive review of background subtraction techniques for mono and multi-sensor surveillance systems that consider different kind of sensors (visible, infrared and audio). The authors presented a different taxonomy by classifying motion detection methods into three categories: per-pixel, per-region, per-frame processing, and from each category it emerges sub-categories. The authors proposed solutions for different challenging situations (adopted from the Wallflower dataset [85]) using the fusion of multi-sensors.

2.2 Comparative papers

In recent years, many studies have also attempted to compare different motion detection methods. The aim of these studies is to define the accuracy, the speed, memory requirements and capabilities to handle several situations. For this purpose, different challenging datasets have been developed in order to give a fair benchmark for all methods. Table 1 summarizes some previous comparison studies, the evaluated methods, datasets and performance metrics used for each comparison.

Table III.1 Comparison studies on motion detection methods

Comparative studies	Tested methods	Datasets used	Metrics used
Toyama *et al.* [86]	Frame difference [86] mean + threshold [86] Mean + covariance (Running Gaussian Average) [14] Mixture of Gaussians [15] Normalized block correlation [87] Temporal derivative (MinMax) [88] Bayesian decision [89] Subspace learning-principle component	Wallflower dataset [85]	False negative (FN) False positive (FP)

	analysis(Eigen-Backgrounds) [42] Linear predictive filter [86] Wallflower method [86]		
Picardi [4]	Running Gaussian average [14] Temporal median filter [90,91] Mixture of Gaussians [15] Kernel density estimation (KDE) [16] Sequential kernel density approximation [92] Subspace learning-principle component analysis(Eigen-Backgrounds) [85] Co-occurrence of image variations [93]	/	Limited accuracy (*L*) Intermediate accuracy (*M*) High accuracy (*H*)
Cheung *et al.* [94]	Frame differencing [86,94] Temporal median filter [90,91] Linear predictive filter [86] Kernel density estimation(KDE)[16] Approximated median filter[94] Kalman filter[95] Mixture of Gaussians[15]	KOGS/-IAKS Universitaet Karlsruhe dataset [96]	Recall Precision
Benezeth *et al.* [97]	Temporal median filter (basic motion detection) [90,91] Running Gaussian average (one Gaussian) [14] Minimum, maximum and maximum Inter-frame difference (MinMax) [88] Mixture of Gaussians [15] Kernel density estimation (KDE) [16] Codebook [19] Subspace learning-principle component analysis (Eigen-Backgrounds) [89]	Synthetic videos Semi-synthetic videos and VSSN 2006 dataset [98] IBM dataset [99]	Recall Precision
Bouwmans [10]	Mixture of Gaussians (MoG)[15] Mixture of Gaussians with particle swarm optimization (MoG-PSO) [100] Improved MoG [101] MoG with MRF [102] MoG improved HLS color space [103] Spatial-Time adaptive per pixel mixture of Gaussian (S-TAP-MoG) [104] Adaptive spatio-temporal neighborhood analysis (ASTNA) [105] Subspace learning-principle component analysis (Eigen-Backgrounds)[42] Subspace learning independent component analysis	Wallflower dataset [85]	False negative (FN) False positive (FP)

	(SL-ICA)[106] Subspace learning incremental non-negative matrix factorization (SL-INMF)[107] Subspace learning using incremental rank-tensor (SL-IRT)[108]		
Goyette *et al.* [109]	Euclidean distance [97] Mahalanobis distance [97] Local-self similarity [110] Mixture of Gaussians (MoG) [15] GMM KaewTraKulPong [58] GMM Zivkovic [60] GMM RECTGAUSS-Tex[111] Bayesian multi-layer[36] ViBe[39] Kernel density estimation(KDE)[16] KDE Nonaka *et al.* [112] KDE Yoshinaga *et al.* [113] Self-organized Background Subtraction (SOBS) [21] Spatially coherent self-organized background subtraction (SC-SOBS) [22] Chebyshev probability [28] ViBe+ [66] Probabilistic super-pixel Markov random fields (PSP-MRF) [114] Pixel-based adaptive segmenter (PBAS)40	CDnet 2012 dataset [115]	Recall Specificity False Positive Rate False Negative Rate Percentage of wrong classifications Precision F-measure
Wang *et al.*[116]	Euclidean distance [97] Mahalanobis distance [97] Multiscale spatio-temp BG Model [117] GMM Zivkovic [60] CP3-online [118] Mixture of Gaussians (MoG) [15] Kernel density estimation(KDE) [16] Spatially coherent self-organized background subtraction (SC-SOBS) [22] K-nearest neighbor method (KNN) [60,119] Fast self-tuning BS [120] Spectral-360 [121] Weightless neural networks (CwisarDH) [51,122] Majority Vote-all [116] Self-balanced local sensitivity (SuBSENSE) [123]	CDnet 2014 dataset [76]	Recall Specificity False Positive Rate False Negative Rate Percentage of wrong classifications Precision F-measure

	Flux tensor with split Gaussian models (FTSG) [124] Majority Vote-3 [116]		
Jodoin *et al.* [126]	Euclidean distance [97] Mahalanobis distance [97] Mixture of Gaussians (MoG) [15] GMM Zivkovic [60] GMM KaewTraKulPong [58] GMM RECTGAUSS-Tex [111] Kernel density estimation (KDE) [16] KDE Nonaka *et al.* [112] KDE Yoshinaga *et al.* [113] Self-organized background subtraction (SOBS) [21] Spatially coherent self-organized background subtraction (SC-SOBS) [22] K-nearest neighbor method (KNN) [119] Spectral-360 [121] Flux tensor with split Gaussian models (FTSG) [124] Pixel-based adaptive segmenter (PBAS) [40] Probabilistic super-pixel Markov random fields (PSP-MRF) [114] Splitting Gaussian mixture model (SGMM) [127] Splitting over-dominating modes GMM (SGMM-SOD) [128] Dirichlet process GMM (DPGMM) [129] Bayesian multi-layer [36] Histogram over time [13] Local-self similarity [110]	CDnet 2012 dataset [115]	False Positive Rate False Negative Rate Percentage of wrong classifications
Bianco *et al.* [130]	IUTIS-1 [130] IUTIS-2 [130] IUTIS-3 [130] Flux tensor with split Gaussian models (FTSG) [124] Self-balanced local sensitivity (SuBSENSE) [123] Weightless neural networks (CwisarDH) [51,122] Spectral-360 [121] fast self-tuning BS [120] K-nearest neighbor method (KNN) [119] Kernel density estimation(KDE) [16]	CDnet 2014 dataset [125]	Recall Specificity False Positive Rate False Negative Rate Percentage of wrong classifications Precision F-measure

	Spatially coherent self-organized background subtraction (SC-SOBS) [22] Euclidean distance [97] Mahalanobis distance [97] Multiscale spatio-temp BG model [117] CP3-online [118] Mixture of Gaussians (MoG) [15] GMM Zivkovic [60] Fuzzy spatial coherence-based SOBS [65] Region-based mixture of Gaussians (RMoG) [131]		
Xu *et al.* [132]	Mixture of Gaussians (MoG) [15] Kernel density estimation(KDE) [16] Codebook [19] Self-organized background subtraction (SOBS) [21] ViBe [39] Pixel-based adaptive segmenter (PBAS) [40] GMM Zivkovic (adaptive GMM) [60] Sample consensus (SACON) [133,134]	CDnet 2014 dataset [125] Video dataset proposed in Wen *et al.* [135]	Recall Specificity False Positive Rate False Negative Rate Percentage of wrong classifications Precision F-measure

The table above shows more than 60 motion detection methods that were tested on 8 datasets. Other datasets exist like: CMU*[136], UCSD†[137], BMC‡[138], SCOVIS[139], MarDCT§[140]. Moreover, new datasets were introduced with depth camera such as: Kinect database**[141] and RGB-D object detection dataset††[142]. We can also find more specialized video datasets such as Fish4knowledge ‡‡[143] for underwater fish detection and tracking. Other comparative studies can be found in [144-148].

Owing to the importance of the motion detection step, it is necessary to examine other motion detection methods that have not been evaluated so far. In particular, various simple modifications to original methods (pre-processing, thresholding, filtering, etc.) can lead to different results.

The objective of this study is to evaluate and compare different motion detection methods and identify the best method for different situations using a challenging complete dataset. To this

* http://www.cs.cmu.edu/~yaser/#software
† http://www.svcl.ucsd.edu/projects/background_subtraction/ucsdbgsub_dataset.htm
‡ http://bmc.iut-auvergne.com/
§ http://www.dis.uniroma1.it/~labrococo/MAR/index.htm
** https://imatge.upc.edu/web/resources/kinect-database-foreground-segmentation
†† http://eis.bristol.ac.uk/~mc13306/
‡‡ http://groups.inf.ed.ac.uk/f4k/index.html

end, we will test the following methods: temporal differencing (frame difference) [86,149,150], three-frame difference (3FD) [151-153], adaptive background (average filter) [90,154,155], forgetting morphological temporal gradient (FMTG) [156], ΣΔ Background estimation [157,158], spatio-temporal Markov random field [159-161], running Gaussian average (RGA) [14,162,163], mixture of Gaussians (MoG) [15,59,164], spatio-temporal entropy image (STEI) [165,166], difference-based spatio-temporal entropy image (DSTEI) [166,167], eigen-background (Eig-Bg) [42,168,169] and simplified self-organized map (Simp-SOBS) [24] methods. Many of these methods (3FD, ΣΔ ,FMTG, STEI, DSTEI, Simp-SOBS) have not been previously evaluated on challenging datasets; to this end, we will use the CDnet2012 [30,115] and CDnet2014 [125] datasets, and compare the results with the well-known and classical algorithms of motion detection in the literature (RGA, MoG and Eig-Bg). The CDnet2014[§§] contains a total of 53 videos of indoor and outdoor scenes with more than 159000 images. Each scene represents different moving objects, such as boats, cars, trucks, cyclists and pedestrians, captured in different scenarios (baseline, shadow, intermittent object motion) as well as under challenging conditions (bad weather, camera jitter, dynamic background and thermal). For each video, a ground truth is provided to allow precise and unified comparison of the change detection methods. Furthermore, the following seven metrics were used for evaluation: recall, specificity, false positive rate, false negative rate, percentage of wrong classification, precision and F-measure.

3. MOTION DETECTION METHODS

Motion detection by a fixed camera poses a major challenge for video surveillance systems in terms of extracting the shape of moving objects. This is due to several problems related to the monitored environment such as: complex backgrounds (e.g. tree leaf movement), weather conditions (e.g. snow or rain) and variations in illumination, in addition to the characteristics of the moving object itself (the similarity of its color to the background color, its size and its distance from the camera). Therefore, in recent years, several methods have been developed to overcome these problems. This section reviews the motion detection algorithms used in this comparative study.

[§§] http://wordpress-jodoin.dmi.usherb.ca/dataset2014/

3.1 Frame Difference (Temporal Differencing)

The definition of frame difference method in a grey-scale image is given in chapter II, section 3.1, to compute the frame difference in this case we use the Eq.II.27. However, in RGB color image, this difference can be computed by various means, such as Manhattan distance (Eq.III.1), Euclidean distance (Eq.III.2), or Chebyshev distance (Eq.III.3) [97,170-173]

$$\zeta(x,y)=\left|I_t^R(x,y)-I_{t-1}^R(x,y)\right|+\left|I_t^G(x,y)-I_{t-1}^G(x,y)\right|+\left|I_t^B(x,y)-I_{t-1}^B(x,y)\right| \quad \text{(III.1)}$$

$$\zeta(x,y)=\sqrt{\left(I_t^R(x,y)-I_{t-1}^R(x,y)\right)^2+\left(I_t^G(x,y)-I_{t-1}^G(x,y)\right)^2+\left(I_t^B(x,y)-I_{t-1}^B(x,y)\right)^2} \quad \text{(III.2)}$$

$$\zeta(x,y)=\max\left\{\left|I_t^R(x,y)-I_{t-1}^R(x,y)\right|,\left|I_t^G(x,y)-I_{t-1}^G(x,y)\right|,\left|I_t^B(x,y)-I_{t-1}^B(x,y)\right|\right\} \quad \text{(III.3)}$$

where $I_t^C(x,y)$ represents the pixel value in the C channel.

In spite of its simplicity, this method offers the following advantages. It exhibits good performance in dynamic environments (e.g. during sunrise or under cloud cover) and works well at the standard video frame rate. In addition, the algorithm is easy to implement, with relatively low design complexity, and can be executed effectively when applied to a real-time system [174].

3.2 Three-frame Difference

The three-frame difference [151] method is based on the temporal differencing method. Two frame difference operations given by (Eq.III.4) and (Eq.III.5) are performed. Then, the results are thresholded using (Eq.III.6) and combined using (Eq.III.7), i.e. the AND logical operator (or the minimum) (see Fig.III.1).

$$\zeta_1(x,y)=\left|I_t(x,y)-I_{t-1}(x,y)\right| \quad \text{(III.4)}$$

$$\zeta_2(x,y)=\left|I_t(x,y)-I_{t+1}(x,y)\right| \quad \text{(III.5)}$$

$$\psi_t(x,y)=\begin{cases}0 & \text{if } \zeta_t(x,y)<Th_t \quad \text{background}\\ 1 & \text{otherwise} \quad \text{foreground}\end{cases} \quad \text{(III.6)}$$

$$\zeta_t(x,y)=Min\left(\psi_1(x,y),\psi_2(x,y)\right) \quad \text{(III.7)}$$

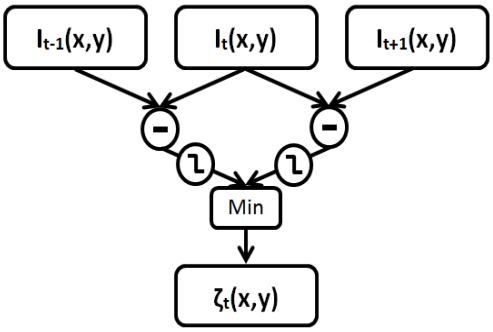


Fig.III.1. Three-frame difference method

This method is robust to noise and provides good detection results for slow moving objects.

3.3 Adaptive Background Subtraction (Running Average Filter)

The concept of this technique is already presented in chapter II, section 3.2. Fig.III.2 summarizes this technique. We should note that the larger the value of the learning rate α, the higher the rate at which the background frame is updated with new changes in the scene. However, α cannot be too large, because it may cause artificial 'tails' behind moving objects [154]. In fact, to prevent tail formation, α must be fixed according to: the observed scene, the size and speed of the moving objects and the distance of these objects from the camera. Furthermore, the problem of continuous movement of small background objects, especially in outdoor scenes (e.g. fluttering flags and swaying tree branches), can be addressed by segmenting such objects with the moving objects.

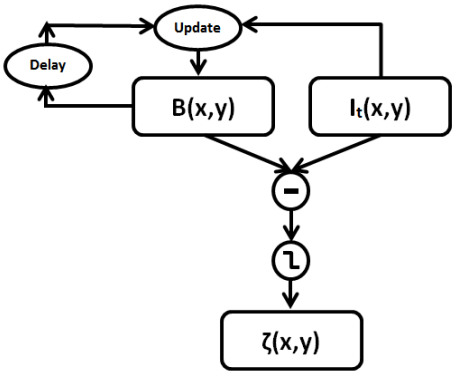


Fig.III.2. Adaptive background detection

3.4 Forgetting Morphological Temporal Gradient

In this method, which was first introduced by Richefeu et al. [156], the difference between temporal dilation and temporal erosion defines the change given by:

$$\delta_\tau\left(I_t(x,y)\right)=\max_{z\in\tau}\left\{I_{t+z}(x,y)\right\} \quad \text{(III.8)}$$

$$\varepsilon_\tau\left(I_t(x,y)\right)=\min_{z\in\tau}\left\{I_{t+z}(x,y)\right\} \quad \text{(III.9)}$$

where $\tau=[t_1,t_2]$ is the temporal structuring element.

In order to reduce, not only, the memory consumption linked to the use of the temporal structuring element but also the sensitivity of this method to large sudden variations, the authors used a running average filter to recursively estimate the values of the temporal erosion and dilation. Thus, (Eq.III.8) and (Eq.III.9) respectively become

$$M_t(x,y)=\alpha I_t(x,y)+(1-\alpha)\max\left\{I_t(x,y),M_{t-1}(x,y)\right\} \quad \text{(III.10)}$$

$$m_t(x,y)=\alpha I_t(x,y)+(1-\alpha)\min\left\{I_t(x,y),m_{t-1}(x,y)\right\} \quad \text{(III.11)}$$

where $M_t(x,y)$ and $m_t(x,y)$ denote respectively the forgetting temporal dilation and the forgetting temporal erosion.

The forgetting morphological temporal gradient is given by

$$\Gamma_t(x,y)=M_t(x,y)-m_t(x,y) \quad \text{(III.12)}$$

Further, the authors have tried to combine this method with the $\Sigma\Delta$ filter to improve the results and automatically define the time constant α.

3.5 $\Sigma\Delta$ Background Estimation

Proposed by Manzanera *et al.* [157], this method is based on the non-linear $\Sigma\Delta$ filter used in electronics applications for analogue-to-digital conversion. The principle of this algorithm is to estimate two values, namely the current background image M_t and the time-variance image V_t, using an iterative process to increment or decrement these values. The algorithm is executed in four steps [158]:

(1) Computation of ΣΔ mean

$$\left.\begin{aligned} &\mathrm{M}_0(x,y) = I_0(x,y) \\ &M_t(x,y) = M_{t-1}(x,y) + \mathrm{sgn}\left(I_t(x,y) - M_{t-1}(x,y)\right) \end{aligned}\right\} \quad \text{(III.13)}$$

(2) Computation of difference

$$\Delta_t(x,y) = \left|M_t(x,y) - I_t(x,y)\right| \quad \text{(III.14)}$$

(3) Computation of ΣΔ variance

$$\left.\begin{aligned} &\mathrm{V}_0(x,y) = \Delta_0(x,y) \\ &\text{if } \Delta_t(x,y) \neq 0, \mathrm{V}_t(x,y) = \mathrm{V}_{t-1}(x,y) + \mathrm{sgn}\left(N \times \Delta_t(x,y) - \mathrm{V}_{t-1}(x,y)\right) \end{aligned}\right\} \quad \text{(III.15)}$$

(4) Computation of motion label

$$\begin{cases} \mathrm{D}_t(\mathrm{x,y})=0 \ \text{ if } \Delta_t(x,y) < \mathrm{V}_t(x,y) \\ \mathrm{D}_t(\mathrm{x,y})=1 \ \text{ else} \end{cases} \quad \text{(III.16)}$$

The only parameter to be set in this method is *N* that represents the amplification factor. However, the application of this method entails several problems such as noise and ghost effects due to moving objects that remain static for long periods in the scene. To overcome this problem, the authors proposed a hybrid geodesic morphological reconstruction filter [157] based on the forgetting morphological operator [156], and given by

$$\Delta_t' = H\ \mathrm{Rec}_{\alpha}^{\Delta_t}\left(Min\left(\left\|\nabla(I_t)\right\|, \left\|\nabla(\Delta_t)\right\|\right)\right) \quad \text{(III.17)}$$

where the gradients of I_t and Δ_t are obtained by convolution with Sobel masks, α is time constant. Further, the classical geodesic relaxation Rec^{Δ_t} is defined by the geodesic dilation as $\mathrm{Rec}^{\Delta_t}\left(Min\left(\left\|\nabla(I_t)\right\|, \left\|\nabla(\Delta_t)\right\|\right)\right) = Min\left(\delta_B\left(Min\left(\left\|\nabla(I_t)\right\|, \left\|\nabla(\Delta_t)\right\|\right), \Delta_t\right)\right)$, where δ is the morphological dilation operator, and *B* is the structuring element.

3.6 Markov Random Field (MRF)-based Motion Detection Algorithm

Introduced by Bouthemy and Lalande [159], this algorithm aims to improve image difference using a Markovian process. To this end, the authors have defined motion detection in images as a binary labelling problem, where the appropriate labels are given by

$$\begin{cases} e(x,y,t) = 1 \text{ if the pixel belongs to a moving object} \\ e(x,y,t) = 0 \text{ if the pixel belongs to a static background} \end{cases} \quad \text{(III.18)}$$

and the observation is the absolute difference between two consecutive frames or between the current frame and a reference image:

$$O_t(x,y) = \left| I_t(x,y) - I_{t-1}(x,y) \right| \tag{III.19}$$

The maximum *a posteriori* (MAP) criterion is used to estimate the appropriate labels of field *E* given field of observation *O* interpreted by maximizing the conditional probability

$$\max_e P\left[E = e \middle| O = o \right] \tag{III.20}$$

Using Bayes' theorem, this is equivalent to

$$\max_e \frac{P\left[O = o \middle| \mathrm{E} = e \right] \mathrm{P}\left[E = e \right]}{P\left[O = o \right]} \tag{III.21}$$

where $P\left[O = o \right]$ is constant with respect to the maximization because the observations are inputs. Conversely, the maximization of $P\left[O = o \middle| \mathrm{E} = e \right] \mathrm{P}\left[E = e \right]$ is equivalent to the minimization of an energy function derived from the Hammersley-Clifford theorem, which states that Markov random fields exhibit a Gibbs distribution with an energy function as follows [175,176]:

$$P\left[E = e \middle| O = o \right] = \frac{e^{-U(o,e)}}{Z} \tag{III.22}$$

where *Z* is a normalizing factor. The energy function *U* is given by the sum of two terms,

$$U(e,o) = U_m(e) + U_a(o,e) \tag{III.23}$$

where U_m denotes the energy that ensures spatio-temporal homogeneity and U_a denotes the adequacy energy that ensures good coherence of the solution compared to the observed data:

$$U_a(o,e) = \frac{1}{2\sigma^2}\left[o_t - \psi(e) \right]^2, \psi(e) = \begin{cases} 0 & \text{if } \left| I_t(x,y) - I_{t-1}(x,y) \right| < Th \\ \alpha & \text{else} \end{cases} \tag{III.24}$$

$$U_m(e) = \sum_{c \in C} V_c(e_s, e_r) \tag{III.25}$$

where *c* denotes a set of binary cliques associated with the chosen neighborhood system describing spatio-temporal interactions between the different pixel intensities [159]. In our case,

there is a 3×3 spatial neighborhood window and two temporal connections: *(x,y,t)* to *(x,y,t-1)* and *(x,y,t)* to *(x,y,t+1)* (see Fig.III.3).

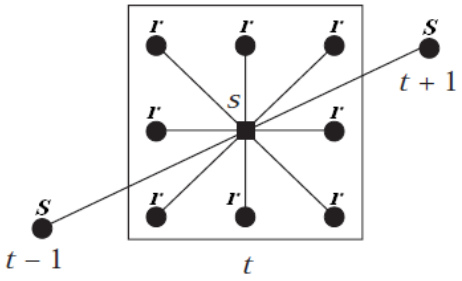


Fig.III.3. 3×3 Spatio-temporal neighborhood [160].

Further, V_c is given by

$$\begin{cases} V_c(e_s,e_r) = V_s(e_s,e_r) + V_p(\mathrm{e}_s^t,\mathrm{e}_s^{t-1}) + V_p(\mathrm{e}_s^t,\mathrm{e}_s^{t+1}) \\ V_s(e_s,e_r) = \begin{cases} -\beta_s & \text{if } \mathrm{e}_s = e_r \\ +\beta_s & \text{if } \mathrm{e}_s \neq e_r \end{cases} \\ V_p(\mathrm{e}_s^t,\mathrm{e}_s^{t-1}) = \begin{cases} -\beta_p & \text{if } \mathrm{e}_s^t = \mathrm{e}_s^{t-1} \\ +\beta_p & \text{if } \mathrm{e}_s^t \neq \mathrm{e}_s^{t-1} \end{cases} \\ V_p(\mathrm{e}_s^t,\mathrm{e}_s^{t+1}) = \begin{cases} -\beta_f & \text{if } \mathrm{e}_s^t = \mathrm{e}_s^{t+1} \\ +\beta_f & \text{if } \mathrm{e}_s^t \neq \mathrm{e}_s^{t+1} \end{cases} \end{cases} \qquad \text{(III.26)}$$

After defining the energy of the Markovian model *U*, the authors considered the problem of minimizing this energy, ultimately using an iterative deterministic relaxation technique [159] (iterated conditional mode method) (see Fig.III.4).

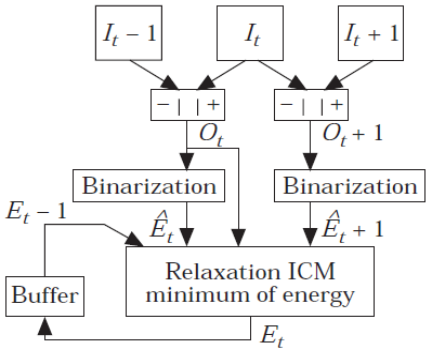


Fig.III.4. MRF-based motion detection [160]

3.7 Running Gaussian Average (One Gaussian)

In this method [14], the background is modelled by fitting a Gaussian distribution (μ,σ) over a histogram for each pixel [4]; this gives the probability density function (pdf) of the background. In order to update this pdf, a running average filter is applied to the parameters of the Gaussian:

$$\mu_t = \alpha I_t + (1-\alpha)\mu_{t-1} \quad \text{(III.27)}$$

$$\sigma_t^2 = \alpha\left(I_t - \mu_t\right)^2 + (1-\alpha)\sigma_{t-1}^2 \quad \text{(III.28)}$$

Then, the pixels correspond to a moving object if the following inequality is satisfied:

$$\left|I_t - \mu_t\right| > D\sigma_t \quad \text{(III.29)}$$

where D is the deviation threshold (e.g. $D = 2.5$).

This method offers the advantages of high execution speed and low memory consumption. However, it suffers from problems associated with the use of the running average (appropriate choices of α and deviation threshold D). Moreover, the use of a single Gaussian to model the background will not give good results for complex backgrounds; in addition, it will favor the extraction of the shadows of moving objects.

3.8 Gaussian Mixture Model (GMM)

In the Gaussian mixture model (or mixture of Gaussians MoG), proposed by Stauffer and Grimson [15], the temporal histogram of each pixel X is modelled using a mixture of K Gaussian distributions in order to precisely model a dynamic background. For example, the periodic or random oscillation of a tree branch that sways in the wind and hides the sun is modelled using two Gaussians. One Gaussian models the temporal variation in the intensity of the pixels when the tree branch obstructs the sun, and the other Gaussian represents the different local intensities produced by the sun. The intensity of each pixel is compared to these Gaussian mixtures, which represent the probability distribution of possible intensities belonging to the dynamic background model. Low probability of belonging to these Gaussian mixtures indicates that the pixel belongs to a moving object. The probability of observing the current pixel value in the multidimensional case is given by

$$P\left(X_t\right)=\sum_{k=1}^{K}\omega_{k,t}\eta\left(\mu_{k,t},\Sigma_{k,t},X_t\right) \tag{III.30}$$

where $\omega_{k,t}$ is the estimated weight associated with the k^{th} Gaussian at time t, $\mu_{k,t}$ is the mean of the k^{th} Gaussian at time t, and $\Sigma_{k,t}$ is the covariance matrix. Further, η is a Gaussian probability density function:

$$\eta\left(\mu_{k,t},\Sigma_{k,t},X_t\right)=\frac{1}{(2\pi)^{\frac{n}{2}}\left|\Sigma\right|^{\frac{1}{2}}}e^{-\frac{1}{2}(X_t-\mu_t)^T\Sigma^{-1}(X_t-\mu_t)} \tag{III.31}$$

Owing to limited memory and computing capacity, the authors have set K in the range of 3 to 5, and they have assumed that the RGB color components are independent and have the same variances. Hence, the covariance matrix is of the form [59]

$$\Sigma_{i,k}=\sigma_{i,k}^2 I \tag{III.32}$$

The first step is to initialize the parameters of the Gaussians (ω_k,μ_k,Σ_k). Then, a test is performed to match each new pixel X with the existing Gaussians.

$$\left|\mu_k-X_t\right|\le D\sigma_k \qquad \text{k=1,...,M} \tag{III.33}$$

where M is the number of Gaussians and D is the deviation threshold.

If a match is found, we update the parameters of this matched Gaussian as follows:

$$\rho=\frac{\alpha}{\omega} \tag{III.34}$$

$$\omega_t=(1-\alpha)\omega_{t-1}+\alpha \tag{III.35}$$

$$\mu_t=\rho X_t+(1-\rho)\mu_{t-1} \tag{III.36}$$

$$\sigma_t^2=\rho\left(X_t-\mu_t\right)^2+(1-\rho)\sigma_{t-1}^2 \tag{III.37}$$

where α and ρ are learning rates; here, ρ is taken from the alternative approximation proposed by Power and Schoonees [177]. For the other unmatched distributions, we maintain their mean and variance; we update only the weights as in (Eq.III.38).

$$\omega_t=(1-\alpha)\omega_{t-1} \tag{III.38}$$

Then, we normalize all the weights as $\omega_k / \sum_{k=1}^{M} \omega_k$.

If no match is found with any of the *K* distributions, we create a new distribution that replaces the parameters of the least probable one, with the current pixel value as its mean, an initially high variance, and low prior weight:

$$\mu_t = X_t \tag{III.39}$$

$$\sigma_t^2 = \sigma_0^2 \text{ (largest value)} \tag{III.40}$$

$$\omega_t = \min(\omega_t) \text{ (smallest value)} \tag{III.41}$$

Then, to distinguish the foreground distribution from the background distribution, we order the distributions by the ratio of their weights to their standard deviations (ω_k / σ_k), assuming that the higher and more compact the distribution, the greater the likelihood of belonging to the background. Then, the first *B* distributions in the ranking order satisfying (Eq.III.42) are considered background [4]:

$$\sum_{k=1}^{B} \omega_k > T \tag{III.42}$$

where *T* is a threshold value. Finally, each new pixel value *X* is compared to these background distributions. If a match is found, this pixel is considered to be a background pixel; otherwise, it is considered to be a foreground pixel.

$$|\mu_k - X_t| \le D\sigma_k \qquad \text{k=1,...,B} \tag{III.43}$$

This method can yield good results for dynamic backgrounds by fitting multiple Gaussians to represent the background more effectively. However, it entails two problems: high computational complexity and parameter initialization. Furthermore, additional parameters must be fixed, such as the threshold value and learning values α and ρ. Many improvements on this method, which deal with the issues and complications affecting the standard algorithm, such as the updating process, initialization, and approximation of the learning rate, can be found in the literature. We refer the readers to the following papers: Power and Schoonees [177] explain in detail the standard mixture of Gaussians (MoG) method used by Stauffer and Grimson; Bouwmans *et al*. [59] discuss the improvements made to the standard MoG method; Carminati and Benois-Pineau [178] use an ISODATA algorithm to estimate the number of K Gaussians for

each pixel, for the matching test the authors use the likelihood maximization followed by Markov regularization instead of the approximation of MAP, Kim *et al.* [179] show that an indoor scene is much closer to a Laplace distribution than to a Gaussian, for which a generalized Gaussian distribution (G-GMM) is proposed instead of a GMM to model the background. Makantasis *et.al* [180] propose to use the Student-t mixture model (STMM) rather than the Gaussian, due to the smaller number of parameters to be tuned. However, the use of STMM increases the complexity of calculation and the memory requirements. To solve this problem, the authors used an image grid, if change is detected using frame difference, the background modelling is applied in the corresponding grid. Many other works tried to use different mixture models like the Dirichlet [129] or hybrid (KDE-GMM) mixture model [78,181].

3.9 Spatio-Temporal Entropy Image

In this method, which was proposed by Ma and Zhang [165], a statistical approach is adopted to measure the variation of each pixel based on its $w \times w$ neighbors along L accumulated frames. A spatio-temporal histogram is created for each pixel, $H_{x,y,q}$ (Fig.III.5), where q denotes the bins of the histogram and the components of the histogram are $\{H_{x,y,1},...,H_{x,y,Q}\}$, where Q is the total number of bins. Then, the corresponding probability density function for each pixel is given by [166]

$$P_{x,y,q} = \frac{H_{x,y,q}}{N} \tag{III.44}$$

where $N = L \times w \times w$.

To determine whether this pixel belongs to the background or foreground, an entropy measure, $E_{x,y}$, is computed from the probability density function.

$$E_{x,y} = -\sum_{q=1}^{Q} P_{x,y,q} \log\left(P_{x,y,q}\right) \tag{III.45}$$

Entropy is a measure of disorder; it is thus assumed that the entropy for a change due to noise is small compared to that due to a moving object.

The diversity of the state of each pixel indicates the intensity of motion at its position [165].

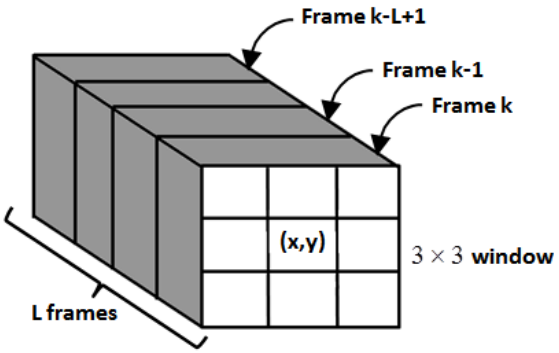


Fig.III.5. Pixels used to construct the spatio-temporal histogram of pixel (i,j)

Ma and Zhang [165] first binarized the entropy result using an adaptive threshold and then applied a morphological filter (close-open operation) to enhance the results.

3.10 Difference-based Spatio-Temporal Entropy Image (DSTEI)

To overcome the problems associated with spatio-temporal entropy, especially the errors resulting from edge pixels, Jing *et al.* [166] attempted to use simple temporal differencing as follows:

$$D_t = \Phi\left(\left|I_t - I_{t-1}\right|\right) \tag{III.46}$$

where Φ quantizes the 256 grey-level values into Q grey levels. As in the case of the STEI method, a spatio-temporal histogram is constructed for each pixel using a $w \times w$ window along a frame difference of L as follows:

$$H_{\mathrm{x,y},q}(L) = \sum_{k=1}^{L} h_{\mathrm{x,y},q}(\mathrm{k}) \tag{III.47}$$

where $h_{\mathrm{x,y},q}(\mathrm{k})$ is the spatial histogram of each pixel in frame k.

To reduce memory consumption in a real-time system, the authors proposed recursive computation of the spatio-temporal histogram using (Eq.III.48), where α is a time constant that determines the influence of the previous frames:

$$H_{\mathrm{x,y},q}(k+1) = \alpha H_{\mathrm{x,y},q}(k) + (1-\alpha) h_{\mathrm{x,y},q}(k+1) \tag{III.48}$$

Then, the pdf of each pixel $P_{x,y,q}$ is obtained by normalizing, (Eq.III.44). Subsequently, the entropy of each pixel is obtained using (Eq.III.45). Finally, a thresholding method is used to extract the motion region.

3.11 Eigen-background Subtraction

The eigen-background method, or subspace learning using principle component learning (SL-PCA), is a background modelling method developed by Olivier *et al.* [42]. The concept underlying this method is that the moving object is rarely found in the same position in the scene across the training frames; hence, its contribution to the eigen-space model is not significant. Conversely, the static objects in the scene can be well described as the sum of various eigen-basis vectors. In this method, an eigen-space is formed using *N* reshaped training frames, $ES = [I_1\ I_2\ ...\ I_N]$, with mean μ and covariance C .

$$\mu = \sum_{k=1}^{N} I_k \tag{III.49}$$

$$C = Cov(ES) = ES \cdot ES^T = \frac{1}{N}\sum_{k=1}^{N}[I_k - \mu]\cdot[I_k - \mu]^T \tag{III.50}$$

Then, we compute *M* principal eigenvectors by PCA, using singular value decomposition or eigen-decomposition; the eigen-decomposition is given by

$$C = V \Lambda V^T \tag{III.51}$$

where $\Lambda = diag\{\lambda_1, \lambda_2, ..., \lambda_N\}$ is the diagonal matrix that contains the eigenvalues ($\lambda_1 > \lambda_2 > ... > \lambda_N$) and V is the eigenvector matrix. Further, V_M consists of the *M* eigenvectors in V that correspond to the *M* largest eigenvalues [169,182]. Then, every new image I is normalized to the mean of the eigen-space μ and projected on these *M* eigenvectors as follows:

$$I' = V_M^T (I - \mu) \tag{III.52}$$

Subsequently, the background is reconstructed by back-projection as follows:

$$B = V_M I' + \mu \tag{III.53}$$

Finally, the foreground can be detected by thresholding the absolute difference as

$$|I - B| > Th \tag{III.54}$$

The authors were very satisfied with the accuracy of the results obtained and specified that their method entails a lower computational load than the mixture of Gaussians method. However, they did not explain how to choose the images that form the eigen-space, because the model is based on the content of these images. If a scene includes a slow or large moving object or crowd movement in the eigen-space, the background model, which should contain only the static objects, will be inappropriate. Furthermore, the authors have not explained how to update the background to consider the possibility of moving objects becoming static.

Many methods have been proposed to overcome these problems, e.g. updating the eigen-background [43,182] and using a selective mechanism [169]. A complete survey on PCA techniques applied to background subtraction can be found in the work of Bouwmans and Zahzah [69,81].

3.12 Simplified Self-organized Background Subtraction (simplified SOBS)

The use of a self-organizing map for background modelling was first proposed by Maddalena and Petrosino [21]. They built a model by mapping each color pixel $I_t(x, y)$ into an $n \times n$ weight vector, thus obtaining a neuronal map B_t of size $[n \times W, n \times H]$, where W and H are the width and height of the observed scene. The initial neuronal map B_0 is obtained in the same manner, where I_0 represents the scene containing the static objects (Fig.III.6).

Then, for each pixel $I_t(x, y)$ from the incoming frame, a matching test is performed with the corresponding weights $b_i, i = 1, ..., n^2$ to find the best match b_m defined by the minimum distance between $I_t(x, y)$ and b_i that should not be greater than a predefined threshold Th

$$d_{\min}(b_m, I_t(x, y)) = \min_{i=1,...,n^2} \{d(b_i, I_t(x, y))\} \leq Th \tag{III.55}$$

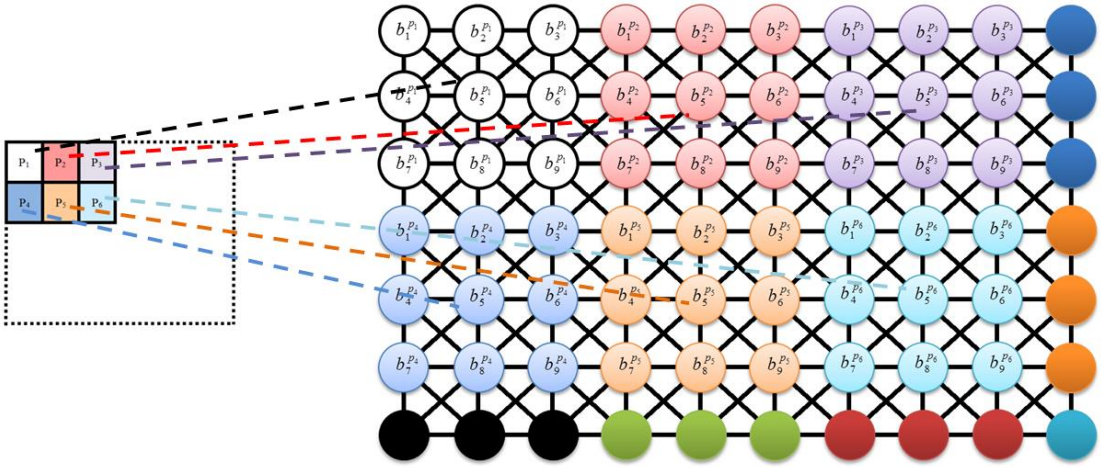


Fig.III.6. (Left) A simple image I_t; (Right) the modelling neuronal map B_t when n = 3

The difference is computed per the color space of the image. Maddalena and Petrosino [21] used the HSV hexagonal cone color space; the Euclidean distance in this case is given by [172]

$$d\left(b_i, I_t(\mathrm{x},\mathrm{y})\right) = \sqrt{\left(\upsilon_{b_i} s_{b_i} \cos\left(h_{b_i}\right) - \upsilon_{I_t} s_{I_t} \cos\left(\mathrm{h}_{I_t}\right)\right)^2 + \left(\upsilon_{b_i} s_{b_i} \sin\left(h_{b_i}\right) - \upsilon_{I_t} s_{I_t} \sin\left(\mathrm{h}_{I_t}\right)\right)^2 + \left(\upsilon_{b_i} - \upsilon_{I_t}\right)^2} \quad \text{(III.56)}$$

If the best match is found in background B_t at position $(\bar{x}, \bar{y})$, we consider the pixel $I_t(x, y)$ to be a background pixel; otherwise, we consider it to be a foreground pixel. We update the model around the best match position as follows:

$$b_{t+1}(i, j) = b_t(i, j) + \alpha_{i,j}(t)\left(I_t(x, y) - b_t(i, j)\right) \quad \text{(III.57)}$$

For $\mathrm{i}=\bar{\mathrm{x}}-\lfloor n/2 \rfloor,\ldots,\bar{\mathrm{x}}+\lfloor n/2 \rfloor$, $\mathrm{j}=\bar{\mathrm{y}}-\lfloor n/2 \rfloor,\ldots,\bar{\mathrm{y}}+\lfloor n/2 \rfloor$, $\alpha_{i,j}(t) = \alpha(\mathrm{t})\omega_{i,j}$, where $\omega_{i,j}$ are $n \times n$ Gaussian weights and $\alpha(\mathrm{t})$ is the learning factor given by

$$\alpha(\mathrm{t}) = \begin{cases} \alpha_1 - t\left(\dfrac{\alpha_1 - \alpha_2}{K}\right), & if \quad 0 \le t \le K \\ \alpha_2, & if \quad t > K \end{cases} \quad \text{(III.58)}$$

where α_1 and α_2 are predefined constants such that $\alpha_2 \le \alpha_1$ and K is the number of frames required for the calibration phase, which depends on how many static initial frames are available for each sequence.

To reduce the computational load as well as the number of parameters to be tuned, Chacon *et.al* [24] proposed a SOM-like architecture in which the mapping is one-to-one, i.e. each neuron

is associated with its corresponding pixel. Since each pixel $I_t(x,y)$ is represented in the HSV color space, each neuron $b(x,y)$ has three inputs h_b, s_b, υ_b, and the matching test is performed as follows:

$$d\left(b, I_t(\mathrm{x},\mathrm{y})\right) \leq Th \quad \wedge \quad \left|\upsilon_{I_t} - \upsilon_b\right| \leq Th_v \tag{III.59}$$

where $d\left(b, I_t(\mathrm{x},\mathrm{y})\right)$ is the Euclidean distance in HSV color space, defined previously in (Eq.III.56), and Th and Th_v are threshold values experimentally set by the authors. The second condition eliminates object shadows. If the result is true, the current pixel is considered to be a background pixel and the weights of the corresponding neuron $b(x,y)$ and its neighbors are updated using (Eq.III.60) and (Eq.III.61), respectively.

$$b_{t+1}(x,y) = b_t(x,y) + \alpha_1\left(I_t(x,y) - b_t(\mathrm{x},y)\right) \tag{III.60}$$

$$b_{t+1}(x',y') = b_t(x',y') + \alpha_2\left(I_t(x',y') - b_t(x',y')\right) \tag{III.61}$$

where $x' = x$ -1, x+1 and $y' = y$ -1, y+1 are the coordinates of the neighboring neurons, and α_1 and α_2 are the learning rates, with $\alpha_1 > \alpha_2$ for non-uniform learning.

If the result of (Eq.III.59) is not true, the current pixel is considered to be a foreground pixel and no update is required. The authors showed that this simplified model performs satisfactorily in different scenarios.

Many other studies have attempted to improve the original self-organized background subtraction method (SOBS). For example, Maddalena and Petrosino [65] introduced fuzzy rules for subtraction and process updates. Further, the authors [183,184] used a 3D neuronal map to model the background; the map consists of *n* layers of the classical 2-grid neuronal map and considers scene changes over time.

4. EVALUATION METRICS AND PERFORMANCE ANALYSIS

To correctly evaluate these methods and achieve a fair comparison, the methods were applied to the same dataset. Further, to define the properties of each method, we used the same seven metrics as those used for CDnet2014: recall, specificity, false positive rate (FPR), false negative rate (FNR), percentage of wrong classification (PWC), precision, and F-measure. The role of

these metrics is to quantify how well each algorithm matches the ground truth. All metrics are based on the following four quantities [79]:

True positives (TP): number of foreground pixels correctly detected

False positives (FP): number of background pixels incorrectly detected as foreground pixels

True negatives (TN): number of background pixels correctly detected

False negatives (FN): number of foreground pixels incorrectly detected as background pixels (also known as misses)

Recall (the sensitivity or true positive rate) is the ratio of the number of foreground pixels correctly detected by the algorithm to the number of foreground pixels in the ground truth [79].

$$\mathrm{Re} = \frac{TP}{TP + FN} \tag{III.62}$$

Specificity (the true negative rate) represents the percentage of correctly classified background pixels.

$$Sp = \frac{TN}{TN + FP} \tag{III.63}$$

False Positive Rate (FPR) is the ratio of the number of background pixels incorrectly detected as foreground pixels by the algorithm to the number of background pixels in the ground truth.

$$FPR = \frac{FP}{TN + FP} \tag{III.64}$$

False Negative Rate (FNR) is the ratio of the number of foreground pixels incorrectly detected as background pixels by the algorithm to the number of background pixels in the ground truth.

$$FNR = \frac{FN}{FN + TP} \tag{III.65}$$

Percentage of Wrong Classification (PWC) is defined as the percentage of wrongly classified pixels.

$$PWC = \frac{FN + FP}{TN + TP + FP + FN} \tag{III.66}$$

Precision is defined as the ratio of the number of foreground pixels correctly detected by the algorithm to the total number of foreground pixels detected by the algorithm [79,94]

$$\Pr = \frac{TP}{TP + FP} \tag{III.67}$$

F-measure or F1 score is a measure of the quality that quantifies, in one scalar value for a frame, the similarity between the resulting foreground detection image and the ground truth [100,59]. Mathematically, it is a trade-off between recall and precision [185]

$$F1 = 2.\frac{\Pr.\mathrm{Re}}{\Pr + \mathrm{Re}} \tag{III.68}$$

For comparison, we have adopted the same approach used on CDnet [109,116] to generate results. First, for each method, we computed all the metrics for each video in each category. Then a category average metric was computed:

$$M_c = \frac{1}{N_c}\sum_v M_{v,c} \tag{III.69}$$

where M_c represents one of the seven metrics (Re, Sp, FPR, FNR, PWC, Pr, F1), N_c is the number of videos in each category, and *v* is a video in category *c*.

We also defined an Overall Average Metric (OAM), which is the simple average of the category averages given by:

$$OAM = \frac{1}{C}\sum_{c=1}^{C} M_c \tag{III.70}$$

where *C* is the number of categories.

To rank all the methods, for each category *c*, we computed the rank of each method for metric *M*. Then, we computed the average rank of this method across all the metrics:

$$RM_c = \frac{1}{7}\sum_{n=1}^{7} Rank(M_c) \tag{III.71}$$

Subsequently, we computed the average over all categories to obtain the average rank across categories *RC* for each method:

$$RC = \frac{1}{C}\sum_{c=1}^{C} RM_c \tag{III.72}$$

In addition, we computed the average rank across the OAM for each method:

$$R = \frac{1}{7}\sum_{n=1}^{7} rank\,(\text{OAM}) \qquad \text{(III.73)}$$

5. RESULTS AND DISCUSSION

We applied each motion detection method described in Section 2 to the CDnet 2014 dataset [125], which includes different scenarios and challenges. Fig.III.7 shows sample frames from each video in each category, while Fig.III.8 shows their corresponding ground truths.

Each motion detection method was followed by an automatic thresholding operation in order to determine region changes and remove small changes in luminosity, except for the RGA, MoG, and MRFMD methods. For RGA and MoG methods, the threshold was fixed to 2.5σ, where σ denotes the standard deviation, and for MRFMD, a fixed threshold $Th = 35$ was used to compute the observation $O(x, y, t)$. We selected Otsu's thresholding method described in chapter II [67].

For the eigen-background method, we set the number of training images to N = 28. These training images were equally spaced by 10 frames and the number of eigen-background vectors was set to M = 3.

For the MoG method, the parameters used were selected in accordance with the work of Nikolov *et al.* [185], who measured the accuracy of the algorithm as a function of each variable parameter. Further, they proposed a set of optimal parameters to improve the performance of the MoG algorithm. Accordingly, we selected the number of Gaussians as K = 3, the learning rate as $\alpha = 0.01$, the foreground threshold as $T = 0.25$, the deviation threshold as $D = 2.5$, and the initial standard deviation as $\sigma_{init} = 20$.Notice that the selected parameters are different from those presented in CDnet2014 [125]. Furthermore, we adopted the approximation of Power and Schoonees [177] to compute the learning rate ρ, hence the values of (μ_t, σ_t) were affected and different too.

For the running Gaussian average method, the learning rate was set to $\alpha = 0.01$and the deviation threshold was set to $D = 2.5$. We note that MoG and RGA were applied to grey-level images in all our tests. We also applied the STEI and DSTEI methods to the grey-level images. To construct the spatio-temporal histogram, we selected a 3×3 window with 5 images and the number of grey levels to Q = 100.

Fig.III.7. Sample frames from each video in each category

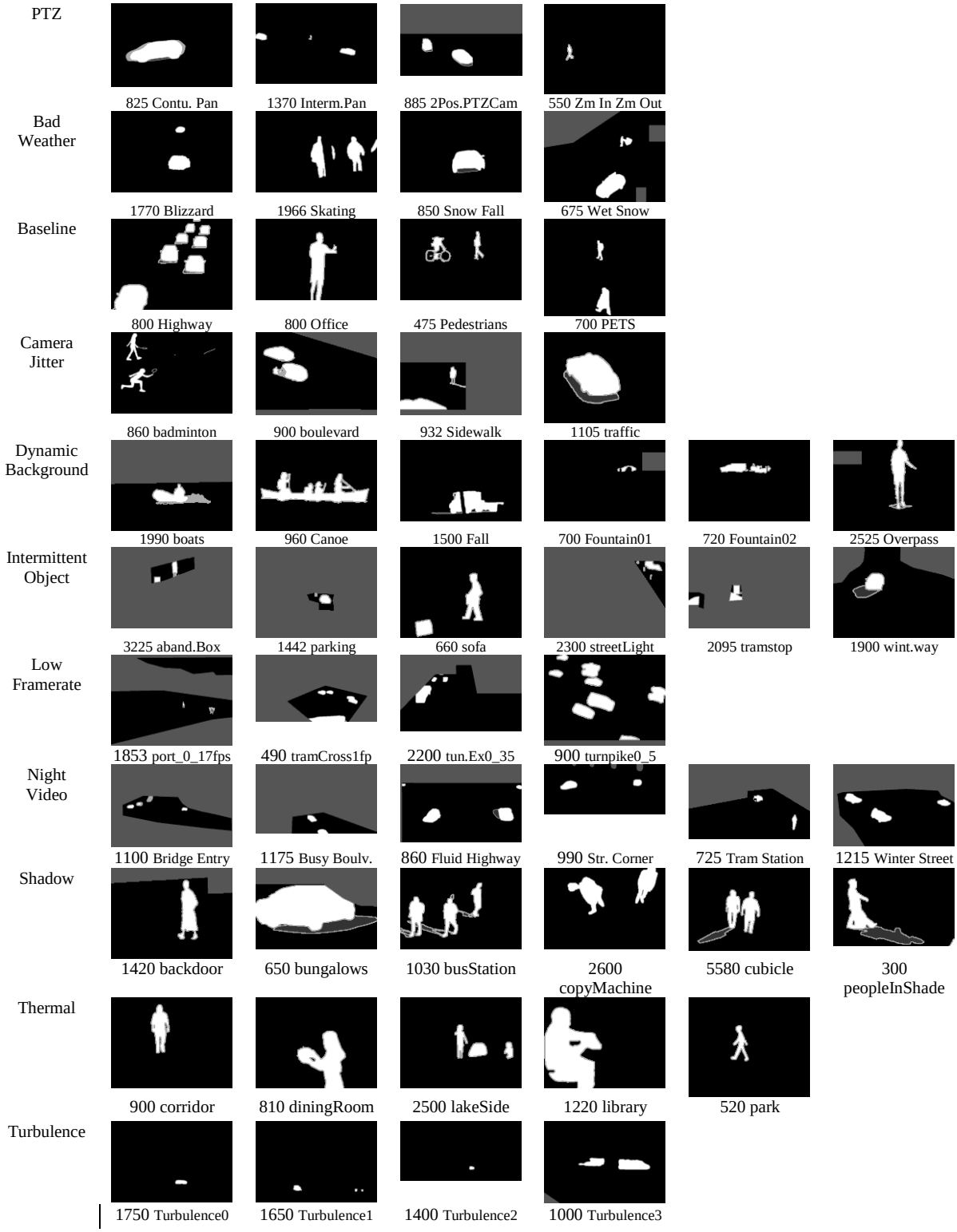


Fig.III.8. Corresponding ground truths of the sample frames in Fig.III.7

For the MRF-based motion detection algorithm, based on the work of Caplier [160], we set the four parameters to $\beta_s = 20, \beta_p = 10, \beta_f = 30, \alpha = 10$. For the $\sum\Delta$ method, the only parameter to be set was *N*; we selected *N* = 3 (typically, *N* = 2, 3, or 4) [158]. For the simplified self-organized background subtraction, the method was tested on the HSV color space, where four parameters, Th, Th_v, α_1 and α_2, had to be set. *Th* and Th_v were set automatically using Otsu's method, and the learning rates were set to α_1=0.02 and $\alpha_2 = 0.01$, according to Ref. [60]. Further, a median filter with *L* = 10 frames was used to initialize the weights of the neuronal map B_0. Finally, for the forgetting morphological temporal gradient method and the adaptive background detection method, the parameter α should take values in the interval $[0,1]$. In our tests for these last two methods, we chose $\alpha = 0.1$. For FD and 3FD, we applied these methods on grayscale images by using Eq.II.27 and Eq.III.4-7 respectively.

Table III.2 summarizes the selected parameters for each tested method.

Table III.2 Selected parameters for each method

Method	Abbrev.	Parameters used
Temporal differencing [7,86]	FD	N/A
Three-frame difference [151,152]	3 FD	N/A
Running average filter [90,154]	RAF	$\alpha = 0.1$
Forgetting morphological temporal gradient [156]	FMTG	$\alpha = 0.1$
$\Sigma\Delta$ background estimation [157]	$\sum\Delta$	N = 3
MRF-based motion detection algorithm [159]	MRFMD	$\beta_s = 20, \beta_p = 10, \beta_f = 30, \alpha = 10$
Spatio-temporal entropy image [165]	STEI	w×w×L = 3×3×5, Q = 100
Difference-based spatio-temporal entropy image [166]	DSTEI	w×w×L = 3×3×5, Q = 100
Running Gaussian average [14]	RGA	$\alpha = 0.01, D = 2.5$
Mixture of Gaussians [15]	MoG	$K = 3, \alpha = 0.01, T = 0.25, D = 2.5$
Eigen-background [42]	Eig-Bg	N = 28, M = 3
Simplified self-organized background subtraction [24]	Simp-SOBS	α_1=0.02 and $\alpha_2 = 0.01$

The overall results of testing these methods using the CDnet dataset (CDnet2012 and CD2014) are reported in Table III.3, where entries are sorted by their average rank across categories (RC).

It is clear from Table III.3 that the STEI method generated poor results compared to the other methods. The use of entropy alone as a metric to detect moving objects did not yield good results because the spatio-temporal accumulation window may contain object edges, which can lead to high diversity (high entropy) and thus impair the segmentation results. Moreover, this error can spread to the entire edge region (see Fig.III.16 and Fig.III.17), generating a very high PWC, high

FPR and very low percentage of correctly classified background pixels (Sp). STEI is also very sensitive (high recall) due to low misses (False negatives, see Fig.III.9).

Table III.3 Overall results across all categories

	Recall	Specificity	FPR	FNR	PWC	Precision	F-Measure	R	R C
Simp-SOBS	0,49362	0,97220	0,02780	0,50638	4,67079	0,51477	0,40097	4,00000	**4,68831**
RGA	0,30123	0,99351	0,00649	0,69877	3,22117	0,49415	0,31465	**3,42857**	5,15584
GMM	0,20606	**0,99593**	0,00407	0,79394	**3,08499**	**0,61021**	0,25420	4,00000	5,44156
Eig-Bg	**0,59669**	0,93715	0,06285	**0,40331**	7,35814	0,41815	**0,41028**	6,57143	6,00000
RAF	0,36107	0,97060	0,02940	0,63893	5,25655	0,44158	0,27924	6,28571	6,05195
FMTG	0,42449	0,95736	0,04264	0,57551	6,37002	0,42101	0,28152	7,14286	6,20779
DSTEI	0,29669	0,96815	0,03185	0,70331	5,63380	0,41881	0,22299	8,14286	6,67532
FD	0,22779	0,97247	0,02753	0,77221	5,43072	0,46649	0,18825	6,85714	6,83117
3FD	0,08117	0,98815	0,01185	0,91883	4,27114	0,46440	0,08201	8,00000	6,94805
MRFMD	0,08693	0,99056	**0,00944**	0,91307	4,02845	0,42689	0,09293	7,00000	7,37662
∑Δ	0,13762	0,98851	0,01149	0,86238	4,11532	0,37271	0,14229	7,42857	7,49351
STEI	0,45870	0,78646	0,21354	0,54130	22,18321	0,12255	0,12881	9,14286	9,12987

Adding the frame difference to this method (DSTEI) increased its precision and decreased the PWC considerably (Fig.III.14, Fig.III.13), except for the "Camera Jitter" category which still has high FPR, PWC and low F-measure (Fig.III.11,13,15). Moreover, from the overall results in Table III.3, we note that DSTEI did not achieve significant improvement over the FD method, owing to the drawbacks of using the spatio-temporal accumulation window and the tails caused by using inappropriate values of α to compute the spatio-temporal histogram recursively (see Fig.III.16 and 17). Notably, DSTEI has acceptable percentage of correctly classified background pixels (Sp) in the "Dynamic Background" category, compared to other methods, see Fig.III.10.

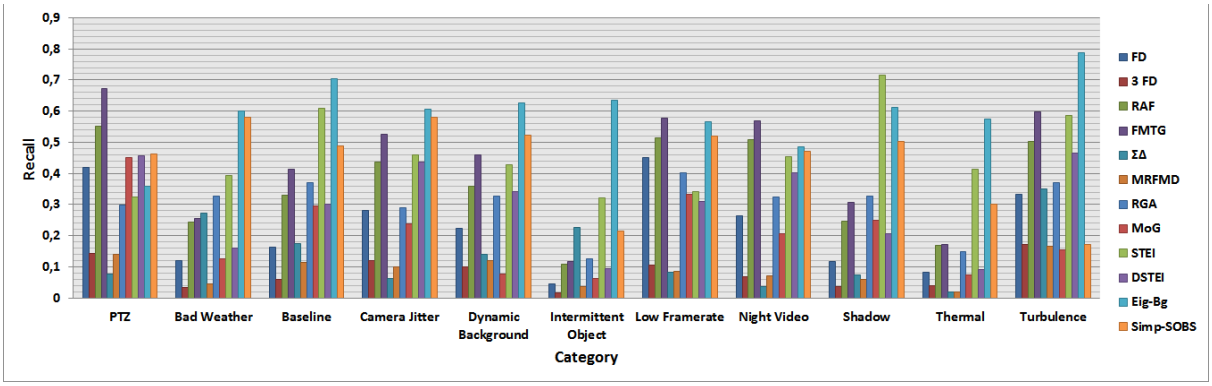


Fig.III.9. Recall results for all tested methods over all categories

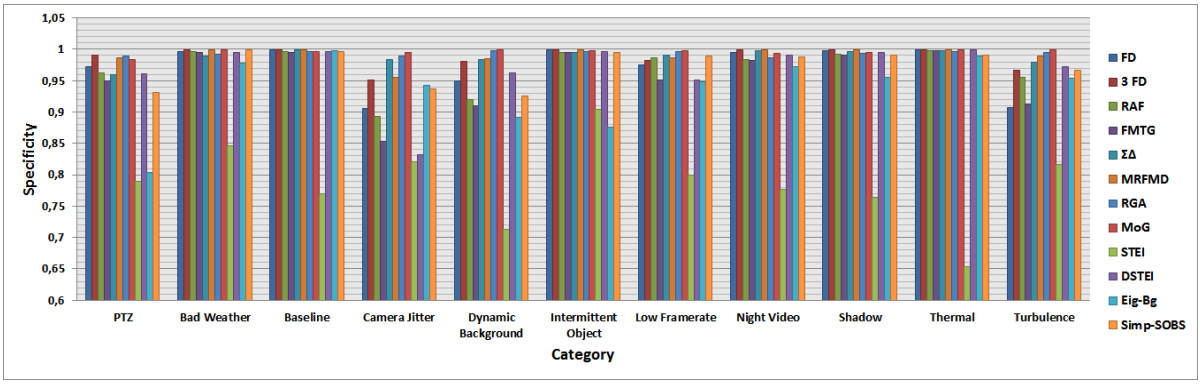


Fig.III.10. Specificity results for all tested methods over all categories

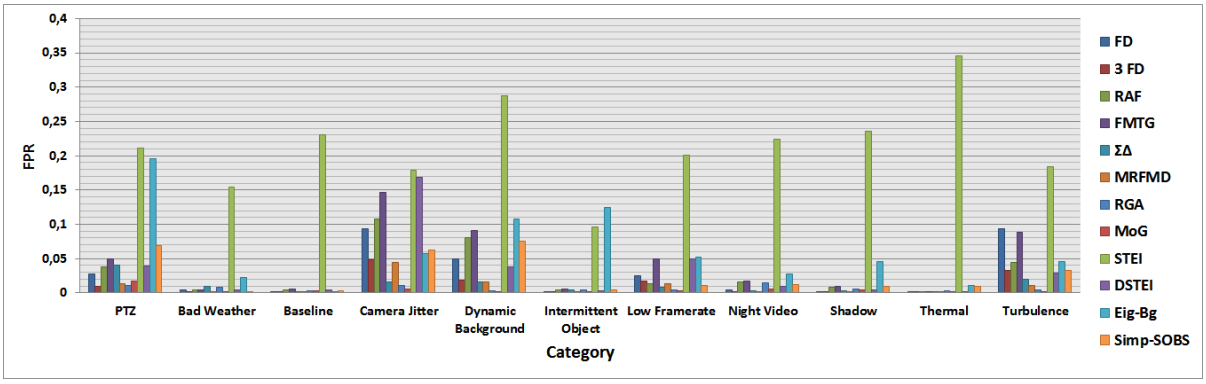


Fig.III.11. False positive rate results for all tested methods over all categories

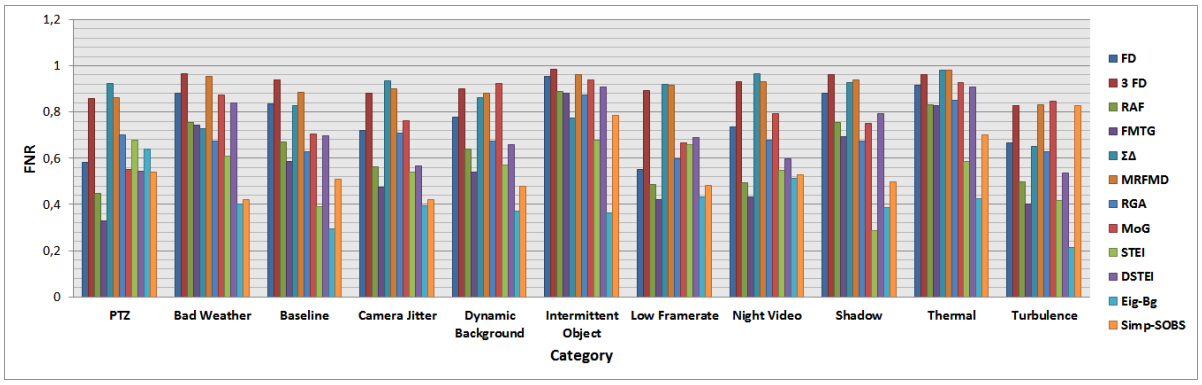


Fig.III.12. False negative rate results for all tested methods over all categories

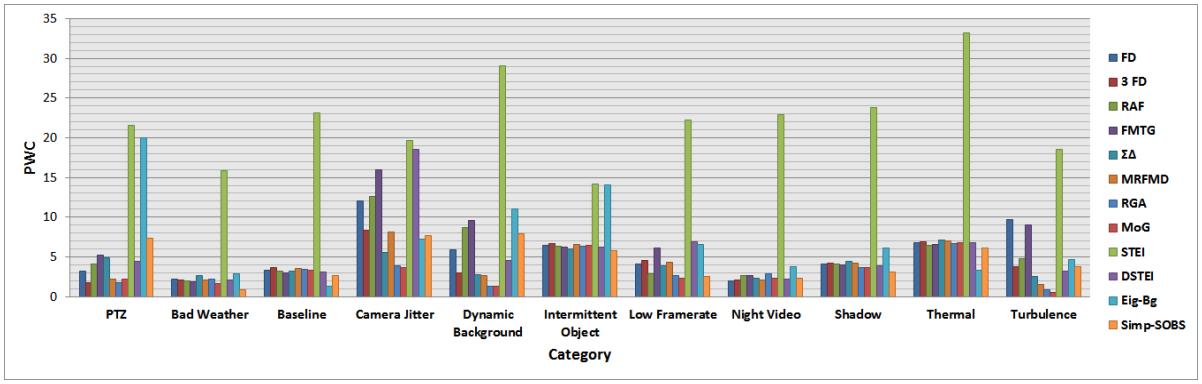


Fig.III.13. Percentage of wrong classification results for all tested methods over all categories

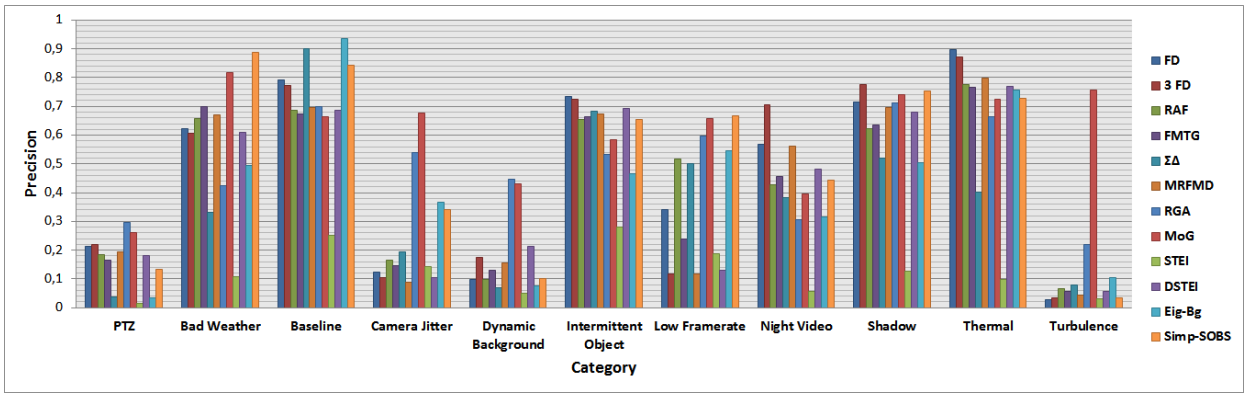


Fig.III.14. Precision results for all tested methods over all categories

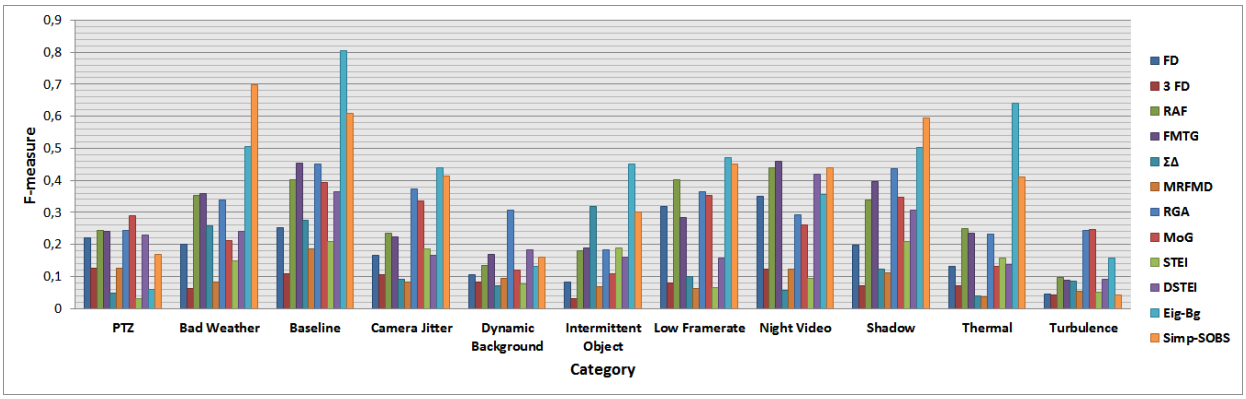


Fig.III.15. F-measure results for all tested methods over all categories

From Table III.3, we can see that the ΣΔ method produced poor results but achieved significant improvement over the STEI method. The ΣΔ method was characterized by a low FPR (Fig.III.11) and high Specificity (Fig.III.10), i.e. many background pixels were correctly classified, but it still suffered from a high false negative rate, especially in "PTZ", "Camera Jitter" and "Thermal" categories, and also low precision in "PTZ", "Bad Weather", "Dynamic Background", "Shadow" and "Thermal" categories (See Tables III.4,5,8 and 12,13).

In, Fig.III.16, we observe that false detections caused by snowfall in the 'Skating' video were eliminated, whereas the objects in motion were not well segmented. We note also that this method has a high precision in the case of "Baseline" category (Fig.III.14). However, results on this category (Fig.III.15) show holes left in the segmented objects (high misses) which produce a low Recall (Fig.III.9, Table III.6). This problem is owed to the initialization step based on temporal differencing.

Table III.4 Pixel-based evaluation of different motion detection methods applied to the "PTZ" category

	Recall	Specificity	FPR	FNR	PWC	Precision	F-Measure	RM_c
GMM	0,44955	0,98300	0,01700	0,55045	2,18306	0,26124	0,28955	3,57143
RGA	0,29935	0,98946	0,01054	0,70065	1,71521	0,29488	0,24364	3,71429
RAF	0,55135	0,96246	0,03754	0,44865	4,10254	0,18547	0,24236	4,42857
3FD	0,14254	0,99055	0,00945	0,85746	1,74923	0,21983	0,12676	5,00000
FD	0,42000	0,97292	0,02708	0,58000	3,23362	0,21225	0,22099	5,28571
FMTG	0,67296	0,94991	0,05009	0,32704	5,22171	0,16603	0,24073	5,85714
DSTEI	0,45699	0,96035	0,03965	0,54301	4,42049	0,18210	0,22979	5,85714
MRFMD	0,13974	0,98662	0,01338	0,86026	2,14460	0,19404	0,12624	6,42857
Simp-SOBS	0,46153	0,93149	0,06851	0,53847	7,29140	0,13466	0,16815	7,42857
Eig-Bg	0,35897	0,80386	0,19614	0,64103	19,98432	0,03295	0,05791	9,71429
ΣΔ	0,07621	0,95886	0,04114	0,92379	4,92846	0,03709	0,04640	9,85714
STEI	0,32390	0,78885	0,21115	0,67610	21,53317	0,01634	0,03047	10,85714

Table III.5 Pixel-based evaluation of different motion detection methods applied to the "bad weather" category

	Recall	Specificity	FPR	FNR	PWC	Precision	F-Measure	RM_c
Simp-SOBS	0,58117	0,99862	0,00138	0,41883	0,87063	0,88652	0,69965	2,14286
GMM	0,12651	0,99966	0,00034	0,87349	1,67592	0,81834	0,21127	4,57143
FMTG	0,25595	0,99546	0,00454	0,74405	1,89175	0,70015	0,35741	5,00000
RAF	0,24383	0,99563	0,00437	0,75617	1,94594	0,65861	0,35216	5,57143
Eig-Bg	0,60037	0,97762	0,02238	0,39963	2,91987	0,49428	0,50623	6,57143
MRFMD	0,04452	0,99863	0,00137	0,95548	2,07388	0,67153	0,08175	6,85714
RGA	0,32721	0,99209	0,00791	0,67279	2,19833	0,42407	0,33892	7,14286
FD	0,12061	0,99620	0,00380	0,87939	2,18648	0,62279	0,19972	7,57143
DSTEI	0,15961	0,99509	0,00491	0,84039	2,11153	0,61012	0,24112	7,57143
3 FD	0,03300	0,99895	0,00105	0,96700	2,09177	0,60550	0,06235	7,71429
ΣΔ	0,27268	0,98994	0,01006	0,72732	2,61408	0,33206	0,25832	8,14286
STEI	0,39261	0,84577	0,15423	0,60739	15,89976	0,10713	0,14777	9,14286

The MRFMD method also yielded poor results, and is similar to the ΣΔ method, with a high FNR and low Recall (Fig.III.12, Fig.III.9), due to its dependence on the initialization step based on the temporal differencing method, leading to incompletely segmented moving objects (holes). Nevertheless, this method performed well by enhancing the image difference in terms of Specificity (Fig.III.10) , PWC (Fig.III.13) and has acceptable precision in the cases of "Bad Weather", "Shadow", "Night Video", and "Thermal" (see Fig.III.14 & Tables III.5,11-13).

Table III.6 Pixel-based evaluation of different motion detection methods applied to the "baseline" category

	Recall	Specificity	FPR	FNR	PWC	Precision	F-Measure	RM_c
Eig-Bg	0,70521	0,99832	0,00168	0,29479	1,32715	0,93552	0,80366	2,14286
Simp-SOBS	0,48942	0,99649	0,00351	0,51058	2,63660	0,84190	0,60852	3,85714
ΣΔ	0,17359	0,99946	0,00054	0,82641	3,23513	0,90046	0,27642	5,42857
RGA	0,37160	0,99672	0,00328	0,62840	3,39816	0,69864	0,44997	5,85714
FMTG	0,41467	0,99483	0,00517	0,58533	3,00519	0,67255	0,45341	6,57143
FD	0,16392	0,99883	0,00117	0,83608	3,35138	0,79205	0,25216	6,71429
RAF	0,33089	0,99618	0,00382	0,66911	3,21227	0,68705	0,40236	7,14286
DSTEI	0,30211	0,99629	0,00371	0,69789	3,09809	0,68536	0,36369	7,42857
3FD	0,06106	0,99947	0,00053	0,93894	3,61961	0,77192	0,10796	7,71429
GMM	0,29563	0,99640	0,00360	0,70437	3,30918	0,66375	0,39360	8,00000
MRFMD	0,11522	0,99872	0,00128	0,88478	3,51801	0,69588	0,18603	8,28571
STEI	0,60964	0,76918	0,23082	0,39036	23,14295	0,25211	0,20983	8,85714

Table III.7 Pixel-based evaluation of different motion detection methods applied to the "Camera Jitter" category

	Recall	Specificity	FPR	FNR	PWC	Precision	F-Measure	RM_c
Eig-Bg	0,6076	0,9421	0,0579	0,3924	7,1837	0,3663	0,4391	3,1429
RGA	0,2905	0,9894	0,0106	0,7095	3,8901	0,5399	0,3741	3,5714
GMM	0,2374	0,9944	0,0056	0,7626	3,6016	0,6779	0,3344	3,7143
Simp-SOBS	0,5808	0,9373	0,0627	0,4192	7,7127	0,3411	0,4147	4,1429
RAF	0,4378	0,8926	0,1074	0,5622	12,6178	0,1664	0,2351	6,8571
FMTG	0,5246	0,8533	0,1467	0,4754	16,0025	0,1457	0,2236	7,0000
ΣΔ	0,0637	0,9837	0,0163	0,9363	5,5428	0,1953	0,0898	7,0000
FD	0,2809	0,9061	0,0939	0,7191	12,0018	0,1227	0,1668	8,1429
3FD	0,1188	0,9513	0,0487	0,8812	8,4001	0,1040	0,1065	8,2857
STEI	0,4609	0,8205	0,1795	0,5391	19,6615	0,1436	0,1872	8,4286
MRFMD	0,0987	0,9554	0,0446	0,9013	8,1616	0,0888	0,0824	8,5714
DSTEI	0,4354	0,8314	0,1686	0,5646	18,5179	0,1041	0,1648	9,1429

In addition, in this method, we had to define the values of the model energy ($\beta_s, \beta_p, \beta_f, \alpha$). Furthermore, the iterative conditional mode (ICM) technique is a suboptimal algorithm that may converge to local minima, but its computational time is considerably shorter than that of a stochastic relaxation scheme (i.e. simulated annealing) [159].

Table III.8 Pixel-based evaluation of different motion detection methods applied to the "Dynamic Background" category

	Recall	Specificity	FPR	FNR	PWC	Precision	F-Measure	RM_c
RGA	0,32604	0,99739	0,00261	0,67396	1,25771	0,44585	0,30555	3,00000
DSTEI	0,34040	0,96264	0,03736	0,65960	4,53876	0,21356	0,18193	5,00000
GMM	0,07816	0,99884	0,00116	0,92184	1,28203	0,43209	0,11900	5,28571
Simp-SOBS	0,52337	0,92515	0,07485	0,47663	7,89821	0,10130	0,16129	5,57143
MRFMD	0,11941	0,98458	0,01542	0,88059	2,65240	0,15423	0,09444	6,14286
FMTG	0,46085	0,90959	0,09041	0,53915	9,53249	0,12996	0,16795	6,42857
RAF	0,35967	0,91939	0,08061	0,64033	8,73891	0,09714	0,13346	7,14286
3FD	0,09905	0,98135	0,01865	0,90095	2,95550	0,17437	0,08254	7,28571
Eig-Bg	0,62761	0,89217	0,10783	0,37239	11,02858	0,07684	0,13219	7,28571
ΣΔ	0,13895	0,98350	0,01650	0,86105	2,75199	0,07037	0,07060	7,57143
FD	0,22397	0,95006	0,04994	0,77603	5,86099	0,09671	0,10502	7,71429
STEI	0,42841	0,71230	0,28770	0,57159	29,06152	0,05091	0,07804	9,57143

Table III.9 Pixel-based evaluation of different motion detection methods applied to the "Intermittent object" category

	Recall	Specificity	FPR	FNR	PWC	Precision	FMeasure	RM_c
ΣΔ	0,22742	0,99530	0,00470	0,77258	6,00318	0,68200	0,31732	4,28571
Simp-SOBS	0,21610	0,99506	0,00494	0,78390	5,81846	0,65376	0,30207	5,42857
DSTEI	0,09327	0,99691	0,00309	0,90673	6,23954	0,69406	0,15943	5,85714
RGA	0,12612	0,99609	0,00391	0,87388	6,33705	0,53360	0,18253	6,28571
FD	0,04566	0,99873	0,00127	0,95434	6,43023	0,73456	0,08313	6,42857
FMTG	0,11714	0,99452	0,00548	0,88286	6,18409	0,66480	0,19018	6,42857
RAF	0,10976	0,99541	0,00459	0,89024	6,31262	0,65421	0,18066	6,71429
Eig-Bg	0,63616	0,87557	0,12443	0,36384	14,03169	0,46714	0,45021	7,00000
3FD	0,01516	0,99954	0,00046	0,98484	6,62883	0,72605	0,02929	7,14286
MRFMD	0,03628	0,99878	0,00122	0,96372	6,57922	0,67419	0,06807	7,28571
GMM	0,06211	0,99820	0,00180	0,93789	6,42132	0,58364	0,10818	7,28571
STEI	0,32070	0,90380	0,09620	0,67930	14,15520	0,27958	0,18753	7,85714

As expected, the frame difference and three-frame difference methods do not show good results, except for high precision, mainly because of the incomplete segmentation of the shape of moving objects, preserving only the edges. Moreover, these methods suffer from overlap of slow moving objects and poor detection of objects far from the camera.

To overcome the problem of incomplete segmentation, the threshold operation in the frame difference method is usually followed by morphological operations to link the edges of the moving objects. Then, regions and holes in the image are filled. Another solution is to combine

frame difference methods with a background subtraction method [5,186]. We also note that these methods demonstrate good detection of foreground pixels in night-time videos compared to background subtraction techniques, see Fig.III.14 and Table III.11, owing to the change of light (vehicle or street light) which impairs the modelled background.

The FMTG yielded acceptable results with observable low FNR (Fig.III.12) and high recall (Fig.III.9) for "PTZ", "Camera Jitter", "Dynamic Background", "Low Framerate", "Night Video" and "Turbulence" categories (Tables III.4,7,8,10,11 and 14).

However, it was characterized by high FPR, caused by artificial tails due to an inappropriate value of α. Moreover, this method was conducive to strong background motion, as in the "Dynamic Background" category, Fig.III.16.

Table III.10 Pixel-based evaluation of different motion detection methods applied to the "Low Framerate" category

	Recall	Specificity	FPR	FNR	PWC	Precision	F-Measure	RM_c
Simp-SOBS	0,51954	0,98959	0,01041	0,48046	2,56486	0,66767	0,44959	2,71429
RGA	0,40264	0,99610	0,00390	0,59736	2,58883	0,59523	0,36304	3,71429
GMM	0,33327	0,99736	0,00264	0,66673	2,31369	0,65832	0,35273	3,71429
RAF	0,51335	0,98607	0,01393	0,48665	2,90861	0,51617	0,40144	4,57143
Eig-Bg	0,56639	0,94749	0,05251	0,43361	6,54513	0,54566	0,46997	5,85714
FD	0,45044	0,97505	0,02495	0,54956	4,09714	0,34237	0,31730	6,42857
FMTG	0,57796	0,95039	0,04961	0,42204	6,06386	0,23980	0,28403	6,57143
ΣΔ	0,08207	0,99134	0,00866	0,91793	3,90652	0,50163	0,10016	7,14286
MRFMD	0,08464	0,98657	0,01343	0,91536	4,31021	0,11816	0,06137	8,85714
3FD	0,10544	0,98281	0,01719	0,89456	4,58337	0,11600	0,08011	9,14286
DSTEI	0,31088	0,95063	0,04937	0,68912	6,88456	0,13012	0,15748	9,28571
STEI	0,34130	0,79972	0,20028	0,65870	22,17830	0,18730	0,06510	10,00000

As with FMTG, the RAF method yielded acceptable results. This result was expected because FMTG is based on a recursive operation of the running average filter, i.e. Eq.II.30. However, FMTG gave good detection results in bad weather conditions (see Fig.III.14, Fig.III.15 & Table III.5) owing to the morphological temporal gradient filter, Eq.III.12. Compared to FD and 3FD, RAF method outperforms them for almost all categories (except night videos) in terms of F-measure, FNR and Sensitivity, but with high FPR.

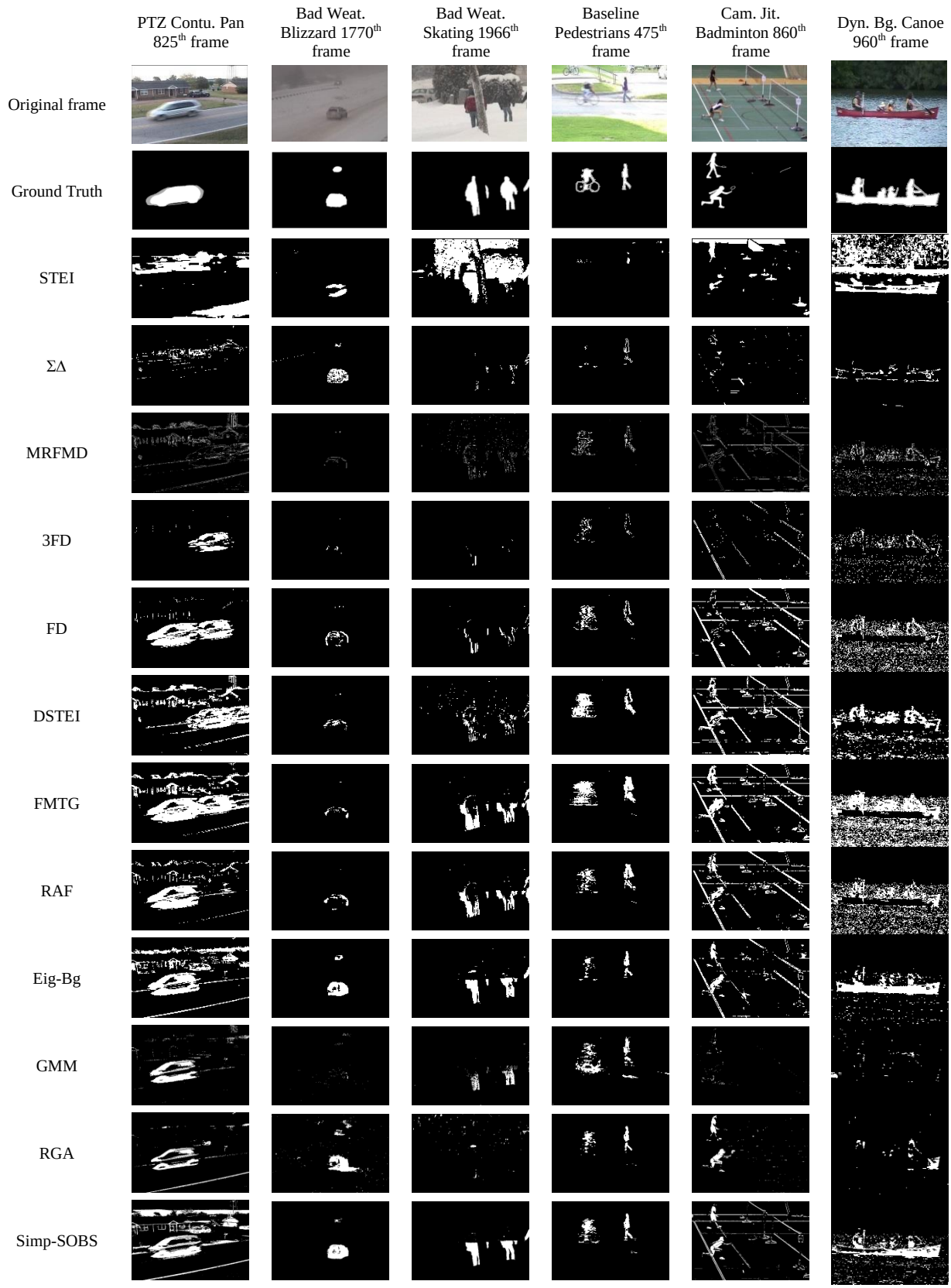
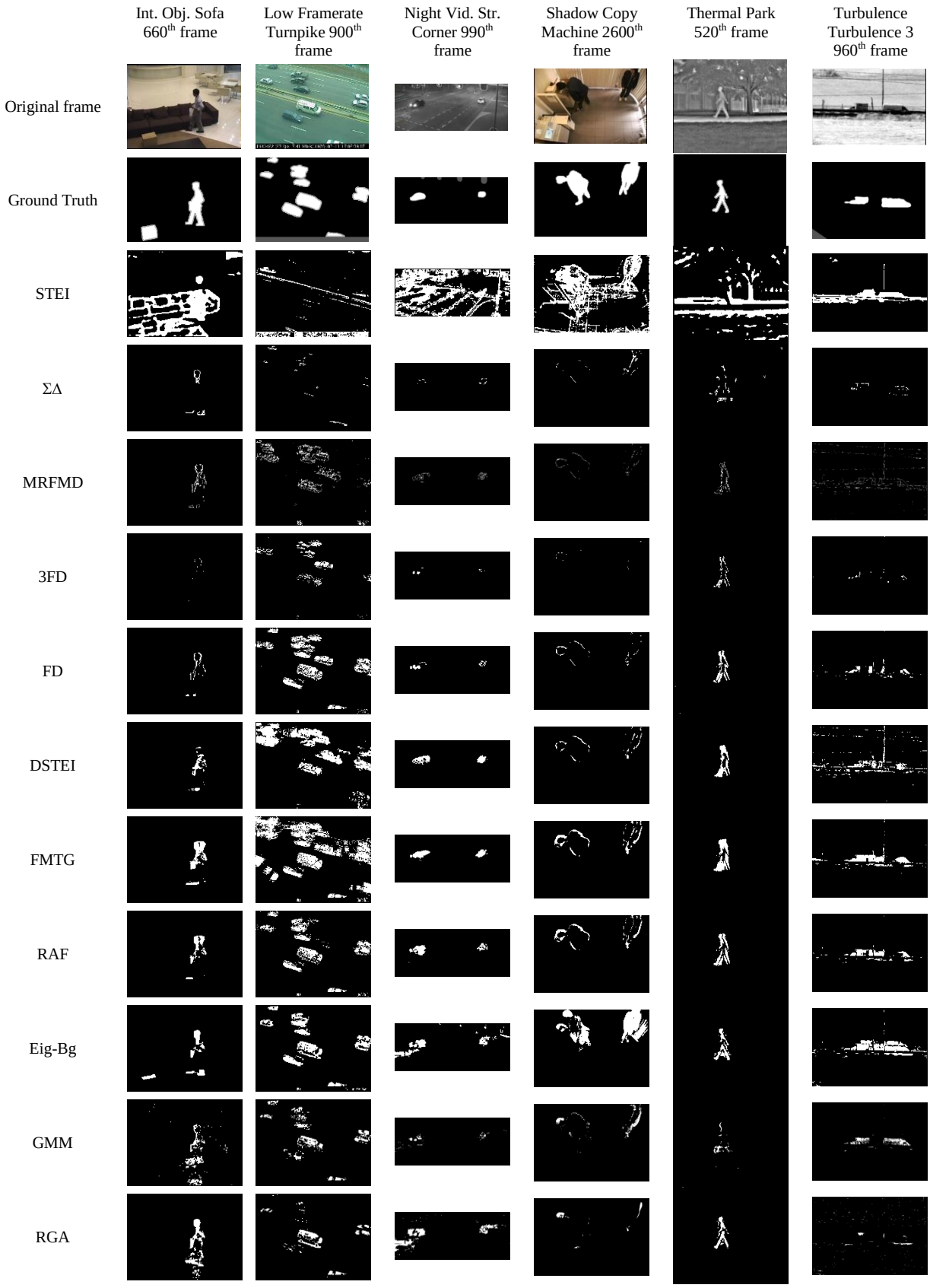


Fig.III.16. Samples from the results of tested motion detection methods (PTZ, Bad Wea., Base.,Cam Jit.,Dyn. Bg)


Int. Obj. Sofa 660th frame
Low Framerate Turnpike 900th frame
Night Vid. Str. Corner 990th frame
Shadow Copy Machine 2600th frame
Thermal Park 520th frame
Turbulence Turbulence 3 960th frame
Original frame
Ground Truth
STEI
ΣΔ
MRFMD
3FD
FD
DSTEI
FMTG
RAF
Eig-Bg
GMM
RGA

Simp-SOBS 

Fig.III.17. Samples from the results of tested motion detection methods (Int.Obj., Low.Fr., N.Vid., Shad., Ther., Turb.)

Table III.11 Pixel-based evaluation of different motion detection methods applied to the "Night video" category

	Recall	Specificity	FPR	FNR	PWC	Precision	F-Measure	RM_c
FD	0,26524	0,99547	0,00453	0,73476	2,01185	0,56769	0,35088	4,71429
DSTEI	0,40097	0,99043	0,00957	0,59903	2,24922	0,48116	0,42043	5,14286
3 FD	0,06949	0,99921	0,00079	0,93051	2,03676	0,70609	0,12229	5,28571
FMTG	0,56808	0,98238	0,01762	0,43192	2,63505	0,45533	0,46018	5,28571
Simp-SOBS	0,47254	0,98834	0,01166	0,52746	2,29129	0,44207	0,43995	5,28571
MRFMD	0,07002	0,99876	0,00124	0,92998	2,07952	0,56291	0,12237	5,57143
RAF	0,50793	0,98401	0,01599	0,49207	2,63158	0,42673	0,43843	5,71429
GMM	0,20595	0,99388	0,00612	0,79405	2,26189	0,39635	0,26142	7,00000
Eig-Bg	0,48550	0,97250	0,02750	0,51450	3,77828	0,31558	0,35541	7,71429
ΣΔ	0,03625	0,99749	0,00251	0,96375	2,27967	0,38100	0,05685	8,14286
RGA	0,32350	0,98592	0,01408	0,67650	2,84418	0,30709	0,29126	8,28571
STEI	0,45413	0,77635	0,22365	0,54587	22,85890	0,05566	0,09322	9,85714

The fourth-best method according to the evaluation results (Table III.3) was the eigen-background method, which yielded good detection results as well as silhouettes of objects. We noted that it had a high Recall among all categories, Fig.III.9 (except "Low Framerate" category), and low FPR, Fig.III.11. For "Intermittent Object" and "PTZ" categories the Eig-Bg shows high FPR and PWC (Fig. III.11,Fig.III.13 and Tables III.4,9). This is due to the absence of update process in the original algorithm. Different techniques have been developed to resolve this problem, For more details see [43,69,81,181].

Remarkably, this method had a great Precision and F-measure for the "Baseline" category which makes this method the ideal approach for easy and mild challenging scenes. However, its results are strongly dependent on the images that form the eigen-space. The presence of moving objects in this space could alter the detection results. The execution time of this method depends on the number of eigenvectors. In our case, the execution time seems acceptable for real-time application. However, memory requirements make it unsuitable for this type of applications.

Table III.12 Pixel-based evaluation of different motion detection methods applied to the "Shadow" category

	Recall	Specificity	FPR	FNR	PWC	Precision	F-Measure	RM_c
Simp-SOBS	0,50198	0,99044	0,00956	0,49802	3,07260	0,75207	0,59353	4,28571
RGA	0,32784	0,99372	0,00628	0,67216	3,61823	0,71265	0,43611	4,57143
GMM	0,24914	0,99518	0,00482	0,75086	3,61844	0,74109	0,34635	5,00000
FD	0,11852	0,99764	0,00236	0,88148	4,08282	0,71392	0,19762	6,14286
DSTEI	0,20761	0,99541	0,00459	0,79239	3,92670	0,67913	0,30582	6,28571
FMTG	0,30569	0,99083	0,00917	0,69431	3,94654	0,63377	0,39529	6,42857
3 FD	0,03776	0,99962	0,00038	0,96224	4,24996	0,77620	0,07113	6,85714
Eig-Bg	0,61292	0,95488	0,04512	0,38708	6,08321	0,50480	0,50255	7,14286
MRFMD	0,06100	0,99894	0,00106	0,93900	4,24903	0,69451	0,11048	7,28571
RAF	0,24641	0,99171	0,00829	0,75359	4,09855	0,62128	0,33949	7,42857
ΣΔ	0,07324	0,99648	0,00352	0,92676	4,39514	0,51865	0,12251	8,28571
STEI	0,71537	0,76417	0,23583	0,28463	23,76070	0,12635	0,20898	8,28571

Table III.13 Pixel-based evaluation of different motion detection methods applied to the "Thermal" category

	Recall	Specificity	FPR	FNR	PWC	Precision	F-Measure	RM_c
Eig-Bg	0,57527	0,98983	0,01017	0,42473	3,34477	0,75660	0,63940	4,71429
RAF	0,16894	0,99785	0,00215	0,83106	6,46159	0,77678	0,24840	4,85714
FD	0,08293	0,99952	0,00048	0,91707	6,76412	0,89736	0,13254	5,28571
FMTG	0,17280	0,99771	0,00229	0,82720	6,53223	0,76712	0,23493	5,42857
Simp-SOBS	0,30052	0,99103	0,00897	0,69948	6,14208	0,72833	0,40935	5,42857
DSTEI	0,09116	0,99884	0,00116	0,90884	6,80556	0,77026	0,13743	6,00000
3FD	0,03946	0,99975	0,00025	0,96054	6,90147	0,87329	0,06998	6,42857
RGA	0,14817	0,99652	0,00348	0,85183	6,69029	0,66375	0,23215	7,14286
MRFMD	0,01977	0,99979	0,00021	0,98023	6,97932	0,79833	0,03642	7,28571
GMM	0,07448	0,99857	0,00143	0,92552	6,79791	0,72397	0,13242	7,57143
STEI	0,41382	0,65387	0,34613	0,58618	33,18293	0,09735	0,15688	8,28571
ΣΔ	0,02021	0,99792	0,00208	0,97979	7,12201	0,40127	0,03848	9,57143

Methods based on Gaussian distributions showed better performance than other methods. The GMM method is characterized by its very high precision and high F1-score in difficult challenging categories ("Bad weather, Dynamic Background, Camera Jitter, Low framerate, Shadow and Turbulence"), and has low PWCs and low FPR in all categories. This is due to the number of K Gaussians used to model the dynamic backgrounds. Notably, the RGA method outperforms the MoG method (Table III.3), with higher Recall (Fig.III.9), low FNR values (Fig.III.12) and higher F-measure (Fig.III.15) in almost all categories. This is owed to the large number of parameters required to set for the MoG algorithm (K, α, T, D, σ), which differ with the challenging conditions presented by a video (day/night, indoor/outdoor, complex/simple

background, with/without noise). Thus, in some cases, the RGA method seemed to be sufficient. Moreover, its computational complexity was lower than that of the MoG method, as was found by Piccardi [4] and Benezeth et al. [97].

Table III.14 Pixel-based evaluation of different motion detection methods applied to the "Turbulence" category

	Recall	Specificity	FPR	FNR	PWC	Precision	F-Measure	RM_c
RGA	0,37053	0,99522	0,00478	0,62947	0,89475	0,22000	0,24396	3,42857
GMM	0,15444	0,99975	0,00025	0,84556	0,46990	0,75562	0,24731	4,14286
Eig-Bg	0,78752	0,95435	0,04565	0,21248	4,71277	0,10395	0,15637	4,71429
Simp-SOBS	0,78287	0,95072	0,04928	0,21713	5,07985	0,11308	0,16392	5,28571
DSTEI	0,46520	0,97167	0,02833	0,53480	3,17950	0,05695	0,09098	5,71429
ΣΔ	0,34950	0,97966	0,02034	0,65050	2,48953	0,07998	0,08646	5,85714
RAF	0,50183	0,95535	0,04465	0,49817	4,79166	0,06753	0,09773	6,14286
MRFMD	0,16698	0,98938	0,01062	0,83302	1,56510	0,04329	0,05261	7,00000
FMTG	0,59875	0,91203	0,08797	0,40125	9,05480	0,05586	0,08904	7,28571
3FD	0,17101	0,96715	0,03285	0,82899	3,76600	0,03516	0,04327	8,57143
STEI	0,58499	0,81651	0,18349	0,41501	18,58043	0,03174	0,05193	9,28571
FD	0,33346	0,90664	0,09336	0,66654	9,71757	0,02896	0,04452	10,57143

Finally, Simp-SOBS showed the best results of all the methods, with high precision, recall, and F-measure, owing to the use of the HSV color space and the condition on the *V* value component that significantly reduced object shadows.

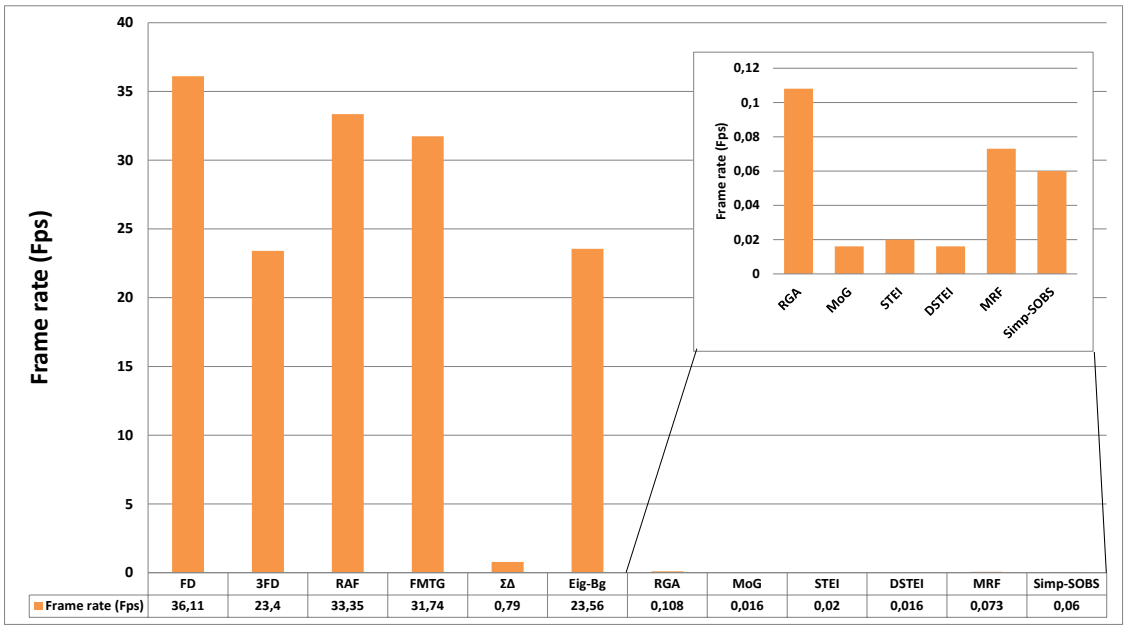
Table III.15 Top three methods for all categories based on the average rank across categories (RC)

	1st	2nd	3rd
PTZ	MoG	RGA	RAF
Bad Weather	Simp-SOBS	MoG	FMTG
Baseline	Eig-Bg	Simp-SOBS	ΣΔ
Camera Jitter	Eig-Bg	RGA	MoG
Dynamic Background	RGA	DSTEI	MoG
Intermittent Object Motion	ΣΔ	Simp-SOBS	DSTEI
Low Framerate	Simp-SOBS	RGA	MoG
Night Video	FD	3FD	DSTEI
Shadow	Simp-SOBS	RGA	MoG
Thermal	Eig-Bg	RAF	FD
Turbulence	RGA	MoG	Eig-Bg

Further, we note from Table III.15 that the results of this method in challenging categories ('Bad weather', 'Low frame rate' and 'Shadow') are as good as those in simple categories ('Baseline'). From the previous results, we can note that the most challenging categories for this

method are: the "Turbulence" category characterized by low Precision (Fig.III.14) and high FNR (Fig.III.12), and the "PTZ" category characterized by high PWC (Fig.III.13) and high FPR (Fig.III.11).

Figure.III.18 shows the frame rate (execution time) of each method, applied on "PETS2006" video from the "Baseline" category, with a resolution of 720×576. Tests were carried on Intel I7 2.3 Ghz with 16 GB RAM, parts of the code was non-vectorized.



	FD	3FD	RAF	FMTG	ΣΔ	Eig-Bg	RGA	MoG	STEI	DSTEI	MRF	Simp-SOBS
Frame rate (Fps)	36,11	23,4	33,35	31,74	0,79	23,56	0,108	0,016	0,02	0,016	0,073	0,06

Fig.III.18. Computational time for each method (presented in frames per second)

The Figure shows that the fastest methods were FD, 3FD, RAF and FMTG because of their simplicity. Eig-Bg shows also a good execution time. This is interpreted by the use of subset of singular values and vectors which overcomes the long time required to compute the N eigenvectors using Eigen decomposition. ΣΔ has a slow frame rate owing to the post processing step required to eliminate the ghost effect. The slowest methods were MoG, STEI and DSTEI. MOG is slow because of the computing complexity linked to the use of K Gaussian distributions, the time required to update their parameters, and to order them. The slowness of STEI and DSTEI is more closely related to the time needed to compute the histogram from the spatio-temporal window. MRF, Simp-SOBS and RGA also had long execution times, but they were not as slow as the previous methods.

6. CONCLUSION

In this chapter, we review and compare motion detection methods using one of the most recent, complete, and challenging datasets: CDnet2012 and CDnet2014. Detailed pixel evaluation was performed using different metrics to enable a user to determine the appropriate method for his or her needs.

From the results reported, we can conclude that there is no ideal method for all situations. Each method performs well in some cases and fails in others. However, it is worth mentioning here that methods based on frame differences are not really designed to detect a complete silhouette of the moving object and thus are underrated here; they aim to detect motion and typically must be combined with other methods to achieve full segmentation. If we had to choose two methods based on the different challenging categories, they would be the simplified self-organized background subtraction [24] (Simp-SOBS) and the running Gaussian average [14] (RGA) methods. The former for the 'Bad weather', 'Baseline', 'intermittent object motion', 'low framerate', 'night video', 'shadow' and 'thermal' categories, and the latter for the 'PTZ', 'camera jitter', 'dynamic background' and 'turbulence' categories. This choice is justified by the high ranks of the methods for nearly all categories based on the average rank across categories as well as their acceptable execution times. The good performance of Simp-SOBS can be explained by the simple but powerful competitive learning used in SOM with an appropriate HSV color space that separates chromaticity from brightness information. The surprising results of RGA are linked to its low complexity with only a few parameters to adjust (e.g. compared to MoG [15]) that was sufficient for most areas of the images tested (only small image portions would have required more complex methods such as MoG), because the whole image background is not always dynamic except for the bad weather condition or maritime applications, where we found that the MoG is superior to the RGA in the tested categories.

Based on this study, a real time implementation on FPGA for classification of moving objects and recognition of actions is presented in the next chapter. In which we detail the different stages from image acquisition, image analysis, classification, behavior understanding to display.

References

[1] I. S. Kim et al., "Intelligent visual surveillance—a survey," Int. J. Control Autom. Syst. 8(5), 926–939 (2010).

[2] M. Paul, S. M. Haque, and S. Chakraborty, "Human detection in surveillance videos and its applications—a review," EURASIP J. Adv. Signal Process. 2013(1), 176 (2013).

[3] W. Hu et al., "A survey on visual surveillance of object motion and behaviors," IEEE Trans. Syst. Man Cybern. Part C 34(3), 334–352 (2004).

[4] M. Piccardi, "Background subtraction techniques: a review," in 2004 IEEE Int. Conf. on Systems, Man and Cybernetics, Vol. 4, pp. 3099–3104 (2004).

[5] D. A. Migliore, M. Matteucci, and M. Naccari, "A revaluation of frame difference in fast and robust motion detection," in Proc. of the 4th ACM Int. Workshop on Video Surveillance and Sensor Networks, pp. 215–218 (2006).

[6] S. Yalamanchili, W. N. Martin, and J. K. Aggarwal, "Extraction of moving object descriptions via differencing," Comput. Graph. Image Process. 18(2), 188–201 (1982).

[7] R. Jain,W. Martin, and J. Aggarwal, "Segmentation through the detection of changes due to motion," Comput. Graph. Image Process. 11(1), 13–34 (1979).

[8] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in Proc. 7th Int. Joint Conf. Artificial Intelligence, Morgan Kaufmann Publishers Inc., San Francisco, California, pp. 674–679 (1981).

[9] B. K. Horn and B. G. Schunck, "Determining optical flow: a retrospective,"Artif. Intell. 59(1–2), 81–87 (1993).

[10] T. Bouwmans, "Recent advanced statistical back ground modeling for foreground detection-a systematic survey," Recent Pat. Comput. Sci. 4(3), 147–176 (2011).

[11] W.-X. Kang, W.-Z. Lai, and X.-B. Meng, "An adaptive background reconstruction algorithm based on inertial filtering," Optoelectron. Lett. 5(6), 468–471 (2009).

[12] S. Jiang and Y. Zhao, "Background extraction algorithm base on Partition Weighed Histogram," in 3rd IEEE Int. Conf. on Network Infrastructure and Digital Content (IC-NIDC 2012), pp. 433–437 (2012).

[13] J. Zheng et al., "Extracting roadway background image: mode-based approach," Transp. Res. Rec. 1944, 82–88 (2006).

[14] C. R. Wren et al., "Pfinder: real-time tracking of the human body," IEEE Trans. Pattern Anal. Mach. Intell. 19(7), 780–785 (1997).

[15] C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," in IEEE Computer Society Conf. on Computer Vision and Pattern Recognition, pp. 246–252 (1999).

[16] 16. A. Elgammal, D. Harwood, and L. Davis, "Non-parametric model for background subtraction," in European Conf. on Computer Vision, pp. 751–767 (2000).

[17] F. El Baf, T. Bouwmans, and B. Vachon, "Type-2 fuzzy mixture of Gaussians model: application to background modeling," in Int. Symp. on Visual Computing, pp. 772–781 (2008).

[18] M. H. Sigari, N. Mozayani, and H. Pourreza, "Fuzzy running average and fuzzy background subtraction: concepts and application," Int. J. Comput. Sci. Network Secur. 8(2), 138–143 (2008).

[19] K. Kim et al., "Real-time foreground–background segmentation using codebook model," Real-Time Imaging 11(3), 172–185 (2005).

[20] M. Xiao, C. Han, and X. Kang, "A background reconstruction for dynamic scenes," in 2006 9th Int. Conf. on Information Fusion, pp. 1–7 (2006).

[21] L. Maddalena and A. Petrosino, "A self-organizing approach to background subtraction for visual surveillance applications," IEEE Trans. Image Process. 17(7), 1168–1177 (2008).

[22] L. Maddalena and A. Petrosino, "The SOBS algorithm: what are the limits?," in IEEE Computer Society Conf. on Computer Vision and Pattern Recognition Workshops (CVPRW 2012), 21–26 (2012).

[23] M. I. Chacon-Murguia and G. Ramirez-Alonso, "Fuzzy-neural selfadapting background modeling with automatic motion analysis for dynamic object detection," Appl. Soft Comput. 36, 570–577 (2015).

[24] M. I. Chacon-Murguia et al., "Simplified SOM-neural model for video segmentation of moving object," in Int. Joint Conf. on Neural Networks, pp. 474–480 (2009).

[25] B. Antic, V. Crnojevic, and D. Culibrk, "Efficient wavelet based detection of moving objects," in 16th Int. Conf. on Digital Signal Processing, pp. 1–6 (2009).

[26] Y.-p. Guan, X.-q. Cheng, and X.-l. Jia, "Motion foreground detection based on wavelet transformation and color ratio difference," in 3rd Int. Congress on Image and Signal Processing (CISP 2010), pp. 1423– 1426 (2010).

[27] S. Messelodi et al., "A Kalman filter based background updating algorithm robust to sharp illumination changes," in Int. Conf. on Image Analysis and Processing, pp. 163–170 (2005).

[28] A. Morde, X. Ma, and S. Guler, "Learning a background model for change detection," in IEEE Computer Society Conf. on Computer Vision and Pattern Recognition Workshops (CVPRW 2012), pp. 15–20 (2012).

[29] G. T. Cinar and J. C. Príncipe, "Adaptive background estimation using an information theoretic cost for hidden state estimation," in Int. Joint Conf. on Neural Networks (IJCNN 2011), pp. 489–494 (2011).

[30] N. Goyette et al., "A novel video dataset for change detection benchmarking," IEEE Trans. Image Process. 23(11), 4663–4679 (2014).

[31] A. F. Bobick and J. W. Davis, "The recognition of human movement using temporal templates," IEEE Trans. Pattern Anal. Mach. Intell. 23(3), 257–267 (2001).

[32] D. Kit, B. Sullivan, and D. Ballard, "Novelty detection using growing neural gas for visuo-spatial memory," in IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS 2011), pp. 1194–1200 (2011).

[33] J. Zhong and S. Sclaroff, "Segmenting foreground objects from a dynamic textured background via a robust Kalman filter," in Proc. Ninth IEEE Int. Conf. on Computer Vision, pp. 44–50 (2003).

[34] F. J. Hernandez-Lopez and M. Rivera, "Change detection by probabilistic segmentation from monocular view," Mach. Vision Appl. 25(5), 1175–1195 (2014).

[35] H. Kim et al., "Robust foreground extraction technique using Gaussian family model and multiple thresholds," in Asian Conf. on Computer Vision 758–768 (2007).

[36] F. Porikli and O. Tuzel, "Bayesian background modeling for foreground detection," in Proc. of the Third ACM Int. Workshop on Video Surveillance & Sensor Networks, pp. 55–58 (2005).

[37] P. Jaikumar, A. Singh, and S. K. Mitra, "Background subtraction in videos using Bayesian learning with motion information," in British Machine Vision Conf. (BMVC), pp. 615–624 (2008).

[38] A. Elgammal et al., "Background and foreground modeling using nonparametric kernel density estimation for visual surveillance," Proc. IEEE 90(7), 1151–1163 (2002).

[39] O. Barnich and M. Van Droogenbroeck, "ViBe: a universal background subtraction algorithm for video sequences," IEEE Trans. Image Process. 20(6), 1709–1724 (2011).

[40] M. Hofmann, P. Tiefenbacher, and G. Rigoll, "Background segmentation with feedback: The pixel-based adaptive segmenter," in IEEE Computer Society Conf. on Computer Vision and Pattern Recognition Workshops (CVPRW 2012), pp. 38–43 (2012).

[41] B. Stenger et al., "Topology free hidden Markov models: application to background modeling," in Proc. Eighth IEEE Int. Conf. on Computer Vision (ICCV 2001), pp. 294–301 (2001).

[42] N. M. Oliver, B. Rosario, and A. P. Pentland, "A Bayesian computer vision system for modeling human interactions," IEEE Trans. Pattern Anal. Mach. Intell. 22(8), 831–843 (2000).

[43] J. Rymel et al., "Adaptive eigen-backgrounds for object detection," in Int. Conf. on Image Processing (ICIP 2004), pp. 1847–1850 (2004).

[44] F. Porikli, "Multiplicative background-foreground estimation under uncontrolled illumination using intrinsic images," in Seventh IEEE Workshops on Application of Computer Vision (WACV/MOTIONS 2005), pp. 20–27 (2005).

[45] C. Zhao, X. Wang, and W.-K. Cham, "Background subtraction via robust dictionary learning," EURASIP J. Image Video Process. 2011(1), 1–12 (2011).

[46] L. Wixson, "Detecting salient motion by accumulating directionallyconsistent flow," IEEE Trans. Pattern Anal. Mach. Intell. 22(8), 774–780 (2000).

[47] X. Lu and R. Manduchi, "Fast image motion segmentation for surveillance applications," Image Vision Comput. 29(2), 104–116 (2011).

[48] D. Zhou and H. Zhang, "Modified GMM background modeling and optical flow for detection of moving objects," in IEEE Int. Conf. on Systems, Man and Cybernetics, pp. 2224–2229 (2005).

[49] L. Cheng et al., "Real-time discriminative background subtraction," IEEE Trans. Image Process. 20(5), 1401–1414 (2011).

[50] B. Han and L. S. Davis, "Density-based multifeature background subtraction with support vector machine," IEEE Trans. Pattern Anal. Mach. Intell. 34(5), 1017–1023 (2012).

[51] M. De Gregorio and M. Giordano, "A WiSARD-based approach to CDnet," in BRICS Congress on Computational Intelligence and 11th Brazilian Congress on Computational Intelligence (BRICS-CCI & CBIC 2013), pp. 172–177 (2013).

[52] X. Cheng et al., "Improving video foreground segmentation with an object-like pool," J. Electron. Imaging 24(2), 023034 (2015).

[53] A. Sobral and A. Vacavant, "A comprehensive review of background subtraction algorithms evaluated with synthetic and real videos," Comput. Vision Image Understanding 122, 4–21 (2014).

[54] A. H. Lai and N. H. Yung, "A fast and accurate scoreboard algorithm for estimating stationary backgrounds in an image sequence," in Proc. of the 1998 IEEE Int. Symp. on Circuits and Systems (ISCAS 1998), Vol. 4, pp. 241–244 (1998).

[55] N. J. McFarlane and C. P. Schofield, "Segmentation and tracking of piglets in images," Mach. Vision Appl. 8(3), 187–193 (1995).

[56] S. Calderara et al., "Reliable background suppression for complex scenes," in Proc. of the 4th ACM Int. Workshop on Video Surveillance and Sensor Networks, pp. 211–214 (2006).

[57] X. Jian et al., "Background subtraction based on a combination of texture, color and intensity," in 9th Int. Conf. on Signal Processing (ICSP 2008), pp. 1400–1405 (2008).

[58] P. KaewTraKulPong and R. Bowden, "An improved adaptive background mixture model for real-time tracking with shadow detection," in Video-Based Surveillance Systems: Computer Vision and Distributed Processing, P. Remagnino et al., Eds., pp. 135–144, Springer US, Boston, Massachusetts (2002).

[59] T. Bouwmans, F. El Baf, and B. Vachon, "Background modeling using mixture of Gaussians for foreground detection-a survey," Recent Pat. Comput. Sci. 1(3), 219–237 (2008).

[60] Z. Zivkovic and F. Van Der Heijden, "Efficient adaptive density estimation per image pixel for the task of background subtraction," Pattern Recognit. Lett. 27(7), 773–780 (2006).

[61] M. Sivabalakrishnan and D. Manjula, "Adaptive background subtraction in dynamic environments using fuzzy logic," Int. J. Comput. Sci. Eng. 2(2), 270–273 (2010).

[62] T. Bouwmans, "Background subtraction for visual surveillance: a fuzzy approach," Handb. Soft Comput. Video Surveillance 5, 103–138 (2012).

[63] Z. Zhao et al., "A fuzzy background modeling approach for motion detection in dynamic backgrounds," in Multimedia and Signal Processing: Second Int. Conf., CMSP 2012, Shanghai, China, December 7-9, 2012. Proc., F. L. Wang et al., Eds., pp. 177–185, Springer Berlin Heidelberg, Berlin, Heidelberg (2012).

[64] D. Culibrk et al., "Neural network approach to background modelling for video object segmentation," IEEE Trans. Neural Networks 18(6), 1614–1627 (2007).

[65] L. Maddalena and A. Petrosino, "A fuzzy spatial coherence-based approach to background/foreground separation for moving object detection," Neural Comput. Appl. 19(2), 179–186 (2010).

[66] Y. Goya et al., "Vehicle trajectories evaluation by static video sensors," in IEEE Intelligent Transportation Systems Conf. (ITSC 2006), pp. 864–869 (2006).

[67] K. Sehairi, F. Chouireb, and J. Meunier, "Comparison study between different automatic threshold algorithms for motion detection," in 4th Int. Conf. on Electrical Engineering (ICEE 2015), pp. 1–8 (2015).

[68] M. Van Droogenbroeck and O. Paquot, "Background subtraction: experiments and improvements for ViBe," in IEEE Computer Society Conf. on Computer Vision and Pattern Recognition Workshops (CVPRW 2012), pp. 32–37 (2012).

[69] T. Bouwmans and E. H. Zahzah, "Robust PCA via principal component pursuit: a review for a comparative evaluation in video surveillance," Comput. Vision Image Understanding 122, 22–34 (2014).

[70] T. Bouwmans et al., "Decomposition into low-rank plus additive matrices for background/foreground separation: a review for a comparative evaluation with a large-scale dataset," Comput. Sci. Rev. 23, 1–71 (2017).

[71] R. Cabral et al., "Unifying nuclear norm and bilinear factorization approaches for low-rank matrix decomposition," in Proc. of the IEEE Int. Conf. on Computer Vision, pp. 2488–2495 (2013).

[72] N. Wang and D.-Y. Yeung, "Bayesian robust matrix factorization for image and video processing," in Proc. of the IEEE Int. Conf. on Computer Vision, pp. 1785–1792 (2013).

[73] C. Guyon, T. Bouwmans, and E.-H. Zahzah, "Foreground detection via robust low rank matrix factorization including spatial constraint with iterative reweighted regression," in 21st Int. Conf. on Pattern Recognition (ICPR 2012), pp. 2805–2808 (2012).

[74] S. Javed et al., "Robust background subtraction to global illumination changes via multiple features-based online robust principal components analysis with Markov random field," J. Electron. Imaging 24(4), 043011 (2015).

[75] B. Zhou, F. Zhang, and L. Peng, "Background modeling for dynamic scenes using tensor decomposition," in 6th Int. Conf. on Electronics Information and Emergency Communication (ICEIEC 2016), pp. 206– 210 (2016).

[76] W. Cao et al., "Total variation regularized tensor RPCA for background subtraction from compressive measurements," IEEE Trans. Image Process. 25(9), 4075–4090 (2016).

[77] A. Sobral et al., "Online stochastic tensor decomposition for background subtraction in multispectral video sequences," in Proc. of the IEEE Int. Conf. on Computer Vision Workshops, pp. 106–113 (2015).

[78] T. Bouwmans, "Traditional and recent approaches in background modeling for foreground detection: an overview," Comput. Sci. Rev. 11, 31–66 (2014).

[79] S. Y. Elhabian, K. M. El-Sayed, and S. H. Ahmed, "Moving object detection in spatial domain using background removal techniques state-of-art," Recent Pat. Comput. Sci. 1(1), 32–54 (2008).

[80] G. Morris and P. Angelov, "Real-time novelty detection in video using background subtraction techniques: state of the art a practical review," in IEEE Int. Conf. on Systems, Man and Cybernetics (SMC 2014), pp. 537–543 (2014).

[81] T. Bouwmans, "Subspace learning for background modeling: a survey," Recent Pat. Comput. Sci. 2(3), 223–234 (2009).

[82] C. Cuevas, R. Martínez, and N. García, "Detection of stationary foreground objects: a survey," Comput. Vision Image Understanding 152, 41–57 (2016).

[83] A. Bux, P. Angelov, and Z. Habib, "Vision based human activity recognition: a review," in Advances in Computational Intelligence Systems: Contributions Presented at the 16th UK Workshop on Computational Intelligence, September 7–9, 2016, Lancaster, UK, P. Angelov et al., Eds., pp. 341–371, Springer International Publishing, Cham (2017).

[84] M. Cristani et al., "Background subtraction for automated multisensory surveillance: a comprehensive review," EURASIP J. Adv. Signal Process. 2010(1), 343057 (2010).

[85] http://research.microsoft.com/en-us/um/people/jckrumm/wallflower/testimages.htm.

[86] K. Toyama et al., "Wallflower: principles and practice of background maintenance," in the Proc. of the Seventh IEEE Int. Conf. on Computer Vision, pp. 255–261 (1999).

[87] T. Matsuyama, T. Ohya, and H. Habe, "Background subtraction for non-stationary scenes," in Proc. of Asian Conf. on Computer Vision, pp. 662–667 (2000).

[88] I. Haritaoglu, D. Harwood, and L. S. Davis, "W4S: a real-time system for detecting and tracking people in 2 1/2D," in European Conf. on Computer Vision, pp. 877–892 (1998).

[89] H. Nakai, "Non-parameterized Bayes decision method for moving object detection," in Proc. Asian Conference on Computer Vision (ACCV), pp. 447–451 (1995).

[90] B. Lo and S. Velastin, "Automatic congestion detection system for underground platforms," in Proc. of 2001 Int. Symp. on Intelligent Multimedia, Video, and Speech Processing, pp. 158–161 (2001).

[91] R. Cucchiara et al., "Detecting moving objects, ghosts, and shadows in video streams," IEEE Trans. Pattern Anal. Mach. Intell. 25(10), 1337– 1342 (2003).

[92] B. Han, D. Comaniciu, and L. Davis, "Sequential kernel density approximation through mode propagation: applications to background modeling," in Asian Conf. on Computer Vision (ACCV 2004), pp. 818– 823 (2004).

[93] M. Seki et al., "Background subtraction based on cooccurrence of image variations," in Proc. IEEE Computer Society Conf. on Computer Vision and Pattern Recognition, pp. 65–72 (2003).

[94] S.-C. S. Cheung and C. Kamath, "Robust background subtraction with foreground validation for urban traffic video," EURASIP J. Adv. Signal Process. 2005(14), 726261 (2005).

[95] K. Karmann and A. Brandt, "Moving object recognition using and adaptive background memory," in Time-Varying Image Processing and Moving Object Recognition, V. Cappellini, Ed., Vol. 2, pp. 289– 307 (1990).

[96] http://i21www.ira.uka.de/image_sequences/.

[97] Y. Benezeth et al., "Comparative study of background subtraction algorithms," J. Electron. Imaging 19(3), 033003 (2010).

[98] http://imagelab.ing.unimore.it/vssn06/.

[99] L. M. Brown et al., "Performance evaluation of surveillance systems under varying conditions," in Proc. IEEE PETS Workshop, pp. 1–8 (2005).

[100] B. White and M. Shah, "Automatically tuning background subtraction parameters using particle swarm optimization," in IEEE Int. Conf. on Multimedia and Expo, pp. 1826–1829 (2007).

[101] Hanzi W. and D. Suter, "A re-evaluation of mixture of Gaussian background modeling [video signal processing applications]," in Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 2005), Vol. 1012 pp. ii/1017–ii/1020 (2005).

[102] K. Schindler and H. Wang, "Smooth foreground-background segmentation for video processing," in Asian Conf. on Computer Vision, pp. 581–590 (2006).

[103] N. A. Setiawan et al., "Gaussian mixture model in improved HLS color space for human silhouette extraction," in Advances in Artificial Reality and Tele-Existence, pp. 732–741, Springer (2006).

[104] M. Cristani and V. Murino, "A spatial sampling mechanism for effective background subtraction," in Proc. 2nd Int. Conf. on Computer Vision Theory and Applications (VISAPP 2007), pp. 403–412 (2007).

[105] M. Cristani and V. Murino, "Background subtraction with adaptive spatio-temporal neighborhood analysis," in Proc. 2nd Int. Conf. on Computer Vision Theory and Applications (VISAPP 2007), pp. 484– 489 (2008).

[106] D.-M. Tsai and S.-C. Lai, "Independent component analysis-based background subtraction for indoor surveillance," IEEE Trans. Image Process. 18(1), 158–167 (2009).

[107] S. S. Bucak, B. Günsel, and O. Gursoy, "Incremental non-negative matrix factorization for dynamic background modelling," in Pattern Recognition in Information Systems (PRIS), pp. 107–116 (2007).

[108] X. Li et al., "Robust foreground segmentation based on two effective background models," in Proc. of the 1st ACM Int. Conf. on Multimedia Information Retrieval, pp. 223–228 (2008).

[109] N. Goyette et al., "Changedetection.net: a new change detection benchmark dataset," in IEEE Computer Society Conf. on Compute Vision and Pattern Recognition Workshops (CVPRW 2012), pp. 1– (2012).

[110] J.-P. Jodoin, G.-A. Bilodeau, and N. Saunier, "Background subtraction based on local shape," arXiv preprint arXiv:1204.6326 (2012).

[111] D. Riahi, P. St-Onge, and G. Bilodeau, "RECTGAUSS-tex: block based background subtraction," in Proce Dept. genie Inform. Genie logiciel, Ecole Polytechn. Montr. Montr., QC, Canada, pp. 1–9 (2012).

[112] Y. Nonaka et al., "Evaluation report of integrated background modelin based on spatio-temporal features," in IEEE Computer Society Conf. on Computer Vision and Pattern Recognition Workshops (CVPRW 2012), pp. 9–14 (2012).

[113] S. Yoshinaga et al., "Background model based on intensity change similarity among pixels," in 19th Korea-Japan Joint Workshop on Frontiers of Computer Vision (FCV 2013), pp. 276–280 (2013).

[114] A. Schick, M. Bäuml, and R. Stiefelhagen, "Improving foreground segmentations with probabilistic superpixel markov random fields," in IEEE Workshop on Change Detection (2012).

[115] http://wordpress-jodoin.dmi.usherb.ca/dataset2012/.

[116] Y. Wang et al., "CDnet 2014: an expanded change detection benchmark dataset," in Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition Workshops, pp. 387–394 (2014).

[117] X. Lu, "A multiscale spatio-temporal background model for motion detection," in IEEE Int. Conf. on Image Processing (ICIP 2014), pp. 3268–3271 (2014).

[118] D. Liang and S. i. Kaneko, "Improvements and experiments of a compact statistical background model," arXiv preprint arXiv:1405.6275 (2014).

[119] B. Tamás, "Detecting and analyzing rowing motion in videos," in BME Scientific Student Conf., Budapest, pp. 1–29 (2016).

[120] B. Wang and P. Dudek, "A fast self-tuning background subtraction algorithm," in Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition Workshops, pp. 395–398 (2014).

[121] M. Sedky, M. Moniri, and C. C. Chibelushi, "Spectral-360: a physics-based technique for change detection," in Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition Workshops, pp. 399–402 (2014).

[122] M. De Gregorio and M. Giordano, "Change detection with weightless neural networks," in Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition Workshops, pp. 403–407 (2014).

[123] P.-L. St-Charles, G.-A. Bilodeau, and R. Bergevin, "Flexible background subtraction with self-balanced local sensitivity," in Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition Workshops, pp. 408–413 (2014).

[124] R. Wang et al., "Static and moving object detection using flux tensor with split Gaussian models," Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition Workshops, pp. 414–418 (2014).

[125] P.-M. Jodoin et al., "Overview and benchmarking of motion detection methods," in Background Modeling and Foreground Detection for Video Surveillance, T. Bouwmans et al., Eds., CRC Press, Boca Raton, Florida (2014).

[126] R. H. Evangelio and T. Sikora, "Complementary background models for the detection of static and moving objects in crowded environments," in 8th IEEE Int. Conf. on Advanced Video and Signal- Based Surveillance (AVSS 2011), pp. 71–76 (2011).

[127] R. H. Evangelio, M. Pätzold, and T. Sikora, "Splitting Gaussians in mixture models," in IEEE Ninth Int. Conf. on Advanced Video and Signal-Based Surveillance (AVSS 2012), pp. 300–305 (2012).

[128] T. S. Haines and T. Xiang, "Background subtraction with dirichlet processes," in European Conf. on Computer Vision, pp. 99–113 (2012).

[129] S. Bianco, G. Ciocca, and R. Schettini, "How far can you get by combining change detection algorithms?," arXiv preprint arXiv:1505.02921 (2015).

[130] S. Varadarajan, P. Miller, and H. Zhou, "Spatial mixture of Gaussians for dynamic background modelling," in 10th IEEE Int. Conf. on Advanced Video and Signal Based Surveillance (AVSS 2013), pp. 63–68 (2013).

[131] http://wordpress-jodoin.dmi.usherb.ca/dataset2014/.

[132] Y. Xu et al., "Background modeling methods in video analysis: a review and comparative evaluation," CAAI Trans. Intell. Technol. 1(1), 43–60 (2016).

[133] H. Wang and D. Suter, "A consensus-based method for tracking: modelling background scenario and foreground appearance," Pattern Recognit. 40(3), 1091–1105 (2007).

[134] H. Wang and D. Suter, "Background subtraction based on a robust consensus method," in 18th Int. Conf. on Pattern Recognition (ICPR 2006), pp. 223–226 (2006).

[135] J. Wen et al., "Joint video frame set division and low-rank decomposition for background subtraction," IEEE Trans. Circuits Syst. Video Technol. 24(12), 2034–2048 (2014).

[136] Y. Sheikh and M. Shah, "Bayesian modeling of dynamic scenes for object detection," IEEE Trans. Pattern Anal. Mach. Intell. 27(11), 1778–1792 (2005).http://www.cs.cmu.edu/~yaser/#software.

[137] V. Mahadevan and N. Vasconcelos, "Spatiotemporal saliency in dynamic scenes," IEEE Trans. Pattern Anal. Mach. Intell. 32(1), 171–177 (2010). http://www.svcl.ucsd.edu/projects/background_subtraction/ucsdbgsub_dataset.htm.

[138] A. Vacavant et al., "A benchmark dataset for outdoor foreground/background extraction," in Asian Conf. on Computer Vision, pp. 291–300 (2012). http://bmc.iut-auvergne.com/.

[139] A. Doulamis et al., "An architecture for a self-configurable video supervision," in Proc. of the 1st ACM Workshop on Analysis and Retrieval of Events/Actions and Workflows in Video streams, pp. 97–104 (2008).

[140] D. D. Bloisi et al., "ARGOS-Venice boat classification," in 12th IEEE Int. Conf. on Advanced Video and Signal Based Surveillance (AVSS 2015), pp. 1–6 (2015). http://www.dis.uniroma1.it/~labrococo/MAR/index.htm.

[141] J. Gallego Vila, "Parametric region-based foreround segmentation in planar and multi-view sequences" (2013). https://imatge.upc.edu/web/resources/kinect-database-foreground-segmentation.

[142] M. Camplani and L. Salgado, "Background foreground segmentation with RGB-D Kinect data: an efficient combination of classifiers," J. Visual Commun. Image Represent. 25(1), 122–136 (2014). http://eis.bristol.ac.uk/~mc13306/.

[143] B. J. Boom et al., "A research tool for long-term and continuous analysis of fish assemblage in coral-reefs using underwater camera footage," Ecol. Inf. 23, 83–97 (2014). http://groups.inf.ed.ac.uk/f4k/index.html.

[144] S. K. Choudhury et al., "An evaluation of background subtraction for object detection vis-a-vis mitigating challenging scenarios," IEEE Access 4, 6133–6150 (2016).

[145] D. H. Parks and S. S. Fels, "Evaluation of background subtraction algorithms with post-processing," in IEEE Fifth Int. Conf. on Advanced Video and Signal Based Surveillance (AVSS 2008), pp. 192–199 (2008).

[146] S. Joudaki, M. S. B. Sunar, and H. Kolivand, "Background subtraction methods in video streams: a review," in 4th Int. Conf. on Interactive Digital Media (ICIDM 2015), pp. 1–6 (2015).

[147] Y. Dhome et al., "A benchmark for background subtraction algorithms in monocular vision: a comparative study," in 2nd Int. Conf. on Image Processing Theory Tools and Applications (IPTA 2010), pp. 66–71 (2010).

[148] D. K. Prasad et al., "Video processing from electro-optical sensors for object detection and tracking in maritime environment: a survey," in IEEE Transactions on Intelligent Transportation Systems, pp. 1–24 (2017).

[149] Z.Wang, J. Xiong, and Q. Zhang, "Motion saliency detection based on temporal difference," J. Electron. Imaging 24(3), 033022 (2015).

[150] S. S. M. Radzi et al., "Extraction of moving objects using frame differencing, ghost and shadow removal," in 5th Int. Conf. on Intelligent Systems, Modelling and Simulation (ISMS 2014), pp. 229–234 (2014).

[151] C. Chen and X. Zhang, "Moving vehicle detection based on union of three-frame difference," in Advances in Electronic Engineering, Communication and Management Vol.2: Proceedings of 2011 International Conference on Electronic Engineering,

Communication and Management (EECM 2011), held on December 24–25, 2011, Beijing, China, D. Jin and Lin S., Eds., pp. 459–464, Springer Berlin Heidelberg, Berlin, Heidelberg (2012).

[152] Z.Wang, "Hardware implementation for a hand recognition system on FPGA," in 5th Int. Conf. on Electronics Information and Emergency Communication (ICEIEC 2015), pp. 34–38 (2015).

[153] Y. Zhang, X. Wang, and B. Qu, "Three-frame difference algorithm research based on mathematical morphology," Proc. Eng. 29, 2705–2709 (2012).

[154] J. Heikkilä and O. Silvén, "A real-time system for monitoring of cyclists and pedestrians," Image Vision Comput. 22(7), 563–570 (2004).

[155] X.-j. Tan, J. Li, and C. Liu, "Avideo-based real-time vehicle detection method by classified background learning," World Trans. Eng. Technol. Edu. 6(1), 189 (2007).

[156] J. Richefeu and A. Manzanera, "A new hybrid differential filter for motion detection," in Computer Vision and Graphics: Int. Conf., ICCVG 2004, Warsaw, Poland, September 2004, Proc., K. Wojciechowsk et al., Eds., pp. 727–732, Springer, Netherlands, Dordrecht (2006).

[157] A. Manzanera and J. C. Richefeu, "A new motion detection algorithm based on $\Sigma - \Delta$ background estimation," Pattern Recognit. Lett. 28(3), 320–328 (2007).

[158] L. Lacassagne et al., "High performance motion detection: some trends toward new embedded architectures for vision systems," J. Real-Time Image Process. 4(2), 127–146 (2009).

[159] P. Bouthemy and P. Lalande, "Recovery of moving object masks in an image sequence using local spatiotemporal contextual information," Opt. Eng. 32(6), 1205–1212 (1993).

[160] A. Caplier, F. Luthon, and C. Dumontier, "Real-time implementations of an MRF-based motion detection algorithm," Real-Time Imaging 4(1), 41–54 (1998).

[161] J. Denoulet et al., "Implementing motion Markov detection on general purpose processor and associative mesh," in Proc. Seventh Int. Workshop on Computer Architecture for Machine Perception (CAMP 2005), pp. 288–293 (2005).

[162] Z. Tang, Z. Miao, and Y.Wan, "Background subtraction using running Gaussian average and frame difference," in Entertainment Computing (ICEC 2007), pp. 411–414, Springer (2007).

[163] S. Jabri et al., "Detection and location of people in video images using adaptive fusion of color and edge information," in 15th Int. Conf. on Pattern Recognition Proc., pp. 627–630 (2000).

[164] J. S. Lumentut and F. E. Gunawan, "Evaluation of recursive background subtraction algorithms for real-time passenger counting at bus rapid transit system," Proc. Comput. Sci. 59, 445–453 (2015).

[165] Y.-F. Ma and H. Zhang, "Detecting motion object by spatio-temporal entropy," in IEEE Int. Conf. Multimedia and Expo, pp. 265–268 (2001).

[166] G. Jing, C. E. Siong, and D. Rajan, "Foreground motion detection by difference-based spatial temporal entropy image," in 2004 IEEE Region 10 Conf. TENCON, pp. 379–382 (2004).

[167] M.-C. Chang and Y.-J. Cheng, "Motion detection by using entropy image and adaptive state-labeling technique," in IEEE Int. Symp. On Circuits and Systems (ISCAS 2007), pp. 3667–3670 (2007).

[168] M. Celenk et al., "Change detection and object tracking in IR surveillance video," in Third Int. IEEE Conf. on Signal-Image Technologies and Internet-Based System (SITIS 2007), pp. 791–797 (2007).

[169] Y. Tian et al., "Selective eigenbackground for background modeling and subtraction in crowded scenes," IEEE Trans. Circuits Syst. Video Technol. 23(11), 1849–1864 (2013).

[170] R. J. Radke et al., "Image change detection algorithms: a systematic survey," IEEE Trans. Image Process. 14(3), 294–307 (2005).

[171] R. Fisher, "Change detection in color images," in Proc. of 7th IEEE Conf. on Computer Vision and Pattern (1999).

[172] N. Selvarasu, A. Nachiappan, and N. Nandhitha, "Euclidean distance based color image segmentation of abnormality detection from pseudo color thermographs," Int. J. Comput. Theory Eng. 2(4), 514–516 (2010).

[173] Q. Zhou and J. K. Aggarwal, "Tracking and classifying moving objects from video," in Proc. of IEEE Workshop on Performance Evaluation of Tracking and Surveillance (2001).

[174] T. Ko, "A survey on behavior analysis in video surveillance for homeland security applications," in 37th IEEE Applied Imagery Pattern Recognition Workshop (AIPR 2008), pp. 1–8 (2008).

[175] S. Geman and D. Geman, "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images," IEEE Trans. Pattern Anal. Mach. Intell. PAMI-6, 721–741 (1984).

[176] R. C. Dubes and A. K. Jain, "Random field models in image analysis," J. Appl. Stat. 16(2), 131–164 (1989).

[177] P.W. Power and J. A. Schoonees, "Understanding background mixture models for foreground segmentation," in Proc. Image and Vision Computing New Zealand, pp. 10–11 (2002).

[178] L. Carminati and J. Benois-Pineau, "Gaussian mixture classification for moving object detection in video surveillance environment," in IEEE Int. Conf. on Image Processing (ICIP 2005), Vol. 3, pp. 113– 116 (2005).

[179] H. Kim et al., "Robust silhouette extraction technique using background subtraction," in 10th Meeting on Image Recognition and Understand (MIRU) (2007).

[180] K. Makantasis, A. Doulamis, and N. F. Matsatsinis, "Student-t background modeling for persons' fall detection through visual cues," in 2012 13th Int. Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS), pp. 1–4 (2012).

[181] Z. Liu, K. Huang, and T. Tan, "Foreground object detection using top-down information based on EM framework," IEEE Trans. Image Process. 21(9), 4204–4217 (2012).

[182] R. Li, C. Yu, and X. Zhang, "Fast robust eigen-background updating for foreground detection," in IEEE Int. Conf. on Image Processing, pp. 1833–1836 (2006).

[183] L. Maddalena and A. Petrosino, "Stopped object detection by learning foreground model in videos," IEEE Trans. Neural Networks Learn. Syst. 24(5), 723–735 (2013).

[184] L. Maddalena and A. Petrosino, "The 3DSOBS+ algorithm for moving object detection," Comput. Vision Image Understanding 122, 65–73 (2014).

[185] B. Nikolov, N. Kostov, and S. Yordanova, “Investigation of mixture of Gaussians method for background subtraction in traffic surveillance,” Int. J. Reasoning-based Intell. Syst. 5(3), 161–168 (2013).

[186] H. Liu et al., “Combining background subtraction and three-frame difference to detect moving object from underwater video,” in OCEANS 2016, Shanghai, pp. 1–5 (2016).

Chapter IV

Real Time Implementation on FPGA of Action Recognition System

“Research is what I’m doing when I don't know what I’m doing”

Wernher von braun

1. INTRODUCTION

In modern society, there is a growing need for technologies such as video surveillance and access control to detect and identify human and vehicle motion in various situations. Intelligent video surveillance attempts to assist human operators when the number of cameras exceeds the operators' capability to monitor them and alerts the operators when abnormal activity is detected. Most intelligent video surveillance systems are designed to detect and recognize human activity. It is difficult to define abnormal activity because there are many behaviors that can represent such activity. Examples include a person entering a subway channel, abandonment of a package, a car running in the opposite direction, and people fighting or rioting. However, it is possible not only to set criteria to detect abnormal activity but also to zoom in on the relevant area to facilitate the work of the operator.

In general, an intelligent video surveillance system has three major stages: detection, classification, and activity recognition [1]. As we presented in chapter I, various methods and systems have been developed to deal with issues in each stage (motion detection [1-7], Object classification [8-11] and action recognition [12-15] such as Advisor system [16] and W^4 system [17]). Additionally, many systems have been also implemented on hardware circuits, FPGA [18-21] or GPU [20,22,23]. Our objective in this chapter is to implement different applications of moving object identification and behavior change detection using feature extraction and motion analysis (finite state machine). Such applications include velocity change detection, direction change detection, and posture change detection. The results can be displayed in the RGB format using chains of parallelized sub-blocks.

We used the high level language the Handel-C and the PixelStreams library of Agility's DK Design Suite to simplify the acquisition and display stages. An RC200E board with an embedded Virtex-II XC2V1000 FPGA was employed for the implementation.

2. OUTLINE OF THE ALGORITHM

2.1 PixelStreams Library

Before we detail and explain our algorithm and the method used to achieve our goals, we should discuss the tools used for our implementation. We used an RC200E board with an embedded XC2V1000 FPGA [24]. This board has multiple video inputs (S-video, camera video, and composite video), multiple video outputs (VGA, S-video, and composite video),

and two ZBT SRAMs, each with a capacity of 2 MB. The language used is Handel-C [25] and the integrated development environment (IDE) is Agility's DK5. This environment is equipped with different platform development kits (PDKs) that include the PixelStreams library [26].

The PixelStreams library is used to develop systems for image and video processing. It includes many blocks (referred to as filters) that perform primary video processing tasks such as acquisition, stream conversion, and filtering. The user has to associate these blocks carefully by indicating the type of the stream (pixel type, coordinate type, and synchronization type). Then, the user can generate the algorithm in Handel-C. Thereafter, he has to add or modify blocks to program his/her method, and finally, he/she must merge the results. It is worth mentioning that these blocks are parameterizable, i.e., we can modify the image processing parameters, such as the size of the acquired image or the threshold. These blocks are fully optimized and parallelized. Fig.IV.1 shows the GUI of PixelStreams.

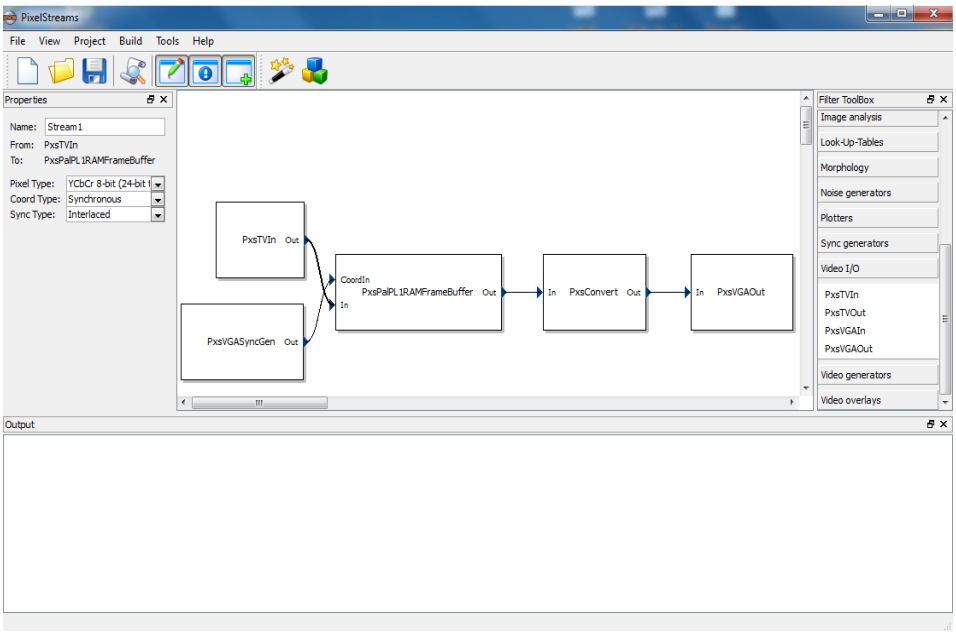


Fig.IV.1. PixelStreams GUI

2.2 Detection Algorithm

We choose to implement the delta frame method for three reasons: its adaptability to changes in luminance, its simplicity, and its low consumption of hardware resources. This method determines the absolute difference between two successive images, and it is executed in two stages: temporal difference and segmentation.

2.2.1 Temporal difference

In this stage, we determine the absolute difference between the previous frame and the current frame as follows.

$$\zeta(x,y) = \left|\frac{dI(x,y)}{dt}\right| = \left|\Delta_{t,t-1}(x,y)\right| = \left|I_t(x,y) - I_{t-1}(x,y)\right| \quad \text{(IV.1)}$$

where $\zeta(x,y)$ is the difference between $I_t(x,y)$ (i.e., the intensity of pixel (x,y) at moment *t*) and $I_{t-1}(x,y)$ (i.e., the intensity of pixel (x,y) at moment *t-1*).

4.2.2 Segmentation

In this stage, significant temporal changes are detected by means of thresholding:

$$\Psi(x,y) = \begin{cases} 0 & \text{if } \zeta(x,y) < \text{Th} \\ 1 & \text{otherwise} \end{cases} \quad \text{(IV.2)}$$

This operation yields a binary card that indicates zones of significant variations in brightness from one image to the other.

2.3 Object Classification

Moving regions detected in video may correspond to different objects in real-world such as pedestrians, vehicles, clutter, etc. It is very important to recognize the type of a detected object in order to track it reliably and analyze its activities. In our case, the technique we used is based on basic recognition and classification method, the criteria height/width ratio. This method is simple but needs the result of detection to be well segmented.

After segmentation, we compute both the width and the height of each detected object, from this ratio we can even specify this moving object whether it is pedestrian or vehicle or other. Fig. IV.2 summarizes the method used for classifying objects.

For each moving object, we will compute the height/width ratio. If this ratio is greater than 1.1, the moving object is a human being, else we will test the ratio again. If it is less than 0.7, the moving object is a vehicle, else it represents another thing. The ratios to specify each class are obtained after testing many scenes. Fig. IV.3 and 4 show the ratio changes for pedestrian and vehicle respectively for a scene of 50 images, in which, we see that the ratio (height/width) does not fall below 1.1 in case of pedestrian and does not exceed 0.7 for vehicle.

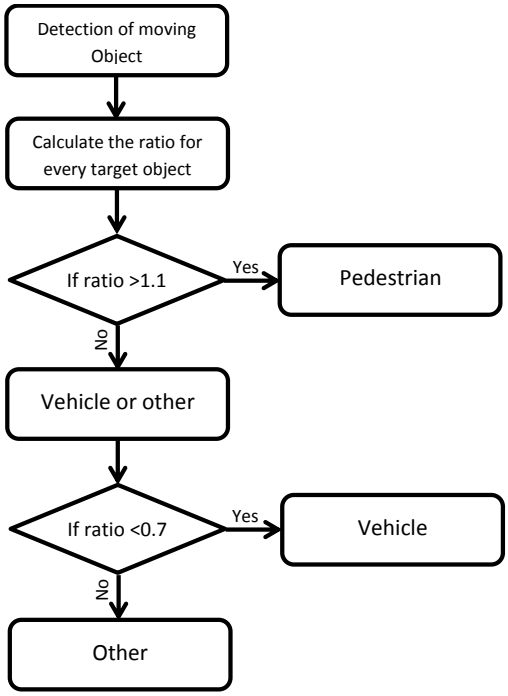


Fig.IV.2. Classification flowchart

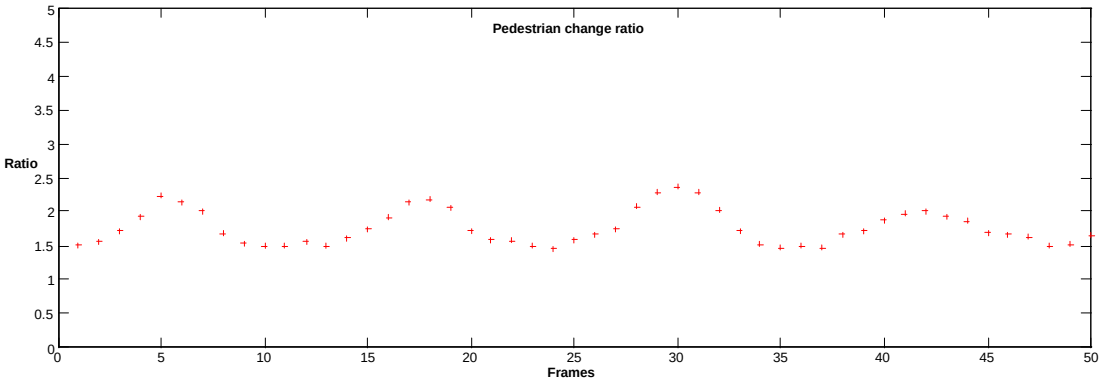


Fig.IV.3. Pedestrian change ratio

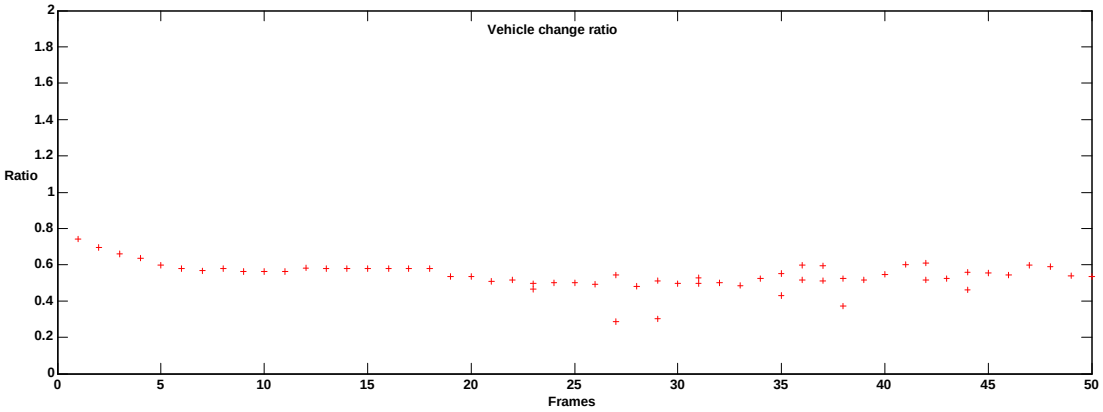


Fig.IV.4. Vehicle change ratio

The proposed algorithm has been tested on several real scenes. Fig. IV.5 shows the background images of two data video used for evaluation. We have chosen different scenes according to illumination condition (Indoor/Outdoor area) and different object in background (trees, leaf, static objects).

Fig.IV.5. Background of the used video dataset

The results of this evaluation were done using Matlab. Fig. IV.6 shows the result of this method, in which we used the dynamic background detection algorithm as motion detection method. We note that the moving objects were well segmented and the classification was accurate.

Fig.IV.6. Object classification

2.4 Feature Extraction and Behavior Change Detection

In this study, simple behavior change detection refers to motion that can be caused by abrupt movements that might represent suspicious actions. To define these actions, we use the parameters of the objects in motion, such as the center of gravity, width, and length. In general, the actions detected by this method are simple yet useful in video surveillance. For example, velocity change detection is useful for detecting a criminal who is being chased by

the police or a car that exceeds the speed limit. Direction change detection is useful for detecting a car that is moving in the wrong direction; and posture change detection is useful for detecting a person who bends to place or pick up an object.

Our implementation involves the following stages: acquisition of the video signal, elimination of noise from the input video signal, detection of moving regions, segmentation for separating the moving objects, extraction of the object parameters, classification of the moving objects, and determining whether movements are suspicious.

2.4.1 Velocity change detection

We can detect suspicious behavior of a person from his/her gait as well as his/her change in velocity near sensitive locations such as banks, airports, and shopping centers. In such cases, we can calculate the speed (in pixels/s) or acceleration (in pixels/s^2) of the suspect in the image space in real time. There are several ways of representing this anomaly: the most widely adopted method in the literature is the use of a bounding box (a rectangle around the suspect).

It is easy to calculate the speed of a moving object. As soon as the speed or acceleration of the object exceeds a certain threshold of normality (predetermined experimentally or on the basis of statistical studies), a bounding box appears around the suspect. However, the issue that needs to be addressed is the calculation of the speed in real-time circuits owing to the absence of mathematical functions (such as square root), types of data (integer or real values), and the object parameters on which we base our calculation.

In general, the speed and acceleration are calculated as follows:

$$velocity\,(t) = (\sqrt{(x_g(t) - x_g(t-dt))^2 + (y_g(t) - y_g(t-dt))^2}) / dt \quad \text{(IV.3)}$$

$$acceleration\,(t) = \big(velocity\,(t) - velocity\,(t-dt)\big) / dt \quad \text{(IV.4)}$$

where $x_g(t), y_g(t)$ and $x_g(t-dt), y_g(t-dt)$ are the co-ordinates of the center of gravity of the object at moments t and $t-dt$, respectively, dt = 40 ms in our case, and $velocity(t)$ and $velocity(t-dt)$ are the velocities of the object at moments t and $t-dt$, respectively.

2.4.2 Direction change detection

Changes in direction or motion in the wrong direction can represent abnormal behavior depending on the situation. For example, roaming around a building or car can be considered as an abrupt cyclic change in direction, possibly indicating the intention of burglary or car theft. Other examples include detection of a car that is moving in the wrong direction or a person who is moving in the opposite direction of a queue at an exit gate or exit corridor in an airport.

To determine the direction, we select parameters that distinguish the object of interest, such as its center of gravity, width, and length. In general, the co-ordinates of the center of gravity can be used to determine whether the object has changed its direction, i.e., whether it has moved rightward or leftward depending on the position of the camera.

The change in direction along the x-axis is given by

$$x_g(t) - x_g(t-dt) \begin{cases} > 0 & \text{The object did not change direction} \\ < 0 & \text{The object changed direction} \end{cases} \quad \text{(IV.5)}$$

The change in direction along the y-axis is given by

$$y_g(t) - y_g(t-dt) \begin{cases} > 0 & \text{The object did not change direction} \\ < 0 & \text{The object changed direction} \end{cases} \quad \text{(IV.6)}$$

These techniques, which are based on the object parameters, can be improved by integrating them with advanced models such as finite state machines (FSMs), see section 5.3.

3. HARDWARE IMPLEMENTATION

Figure.IV.7 shows the general outline of our FPGA implementation.

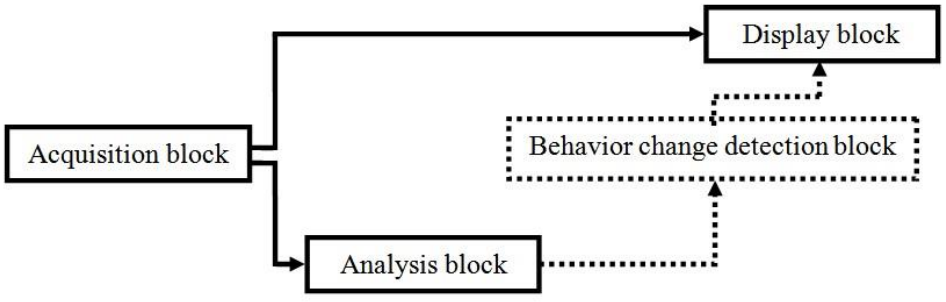


Fig.IV.7. General outline of behavior change detection

This general outline consists of four blocks: an acquisition block, an analysis block, a display block, and an intermediate block between the display block and the analysis block, which is the Behavior change detection block.

3.1 Acquisition Block

Acquisition is achieved using the standard camera associated with the RC200E board. The video input processor, Philips SAA7113H, acquires the frames in the PAL format at a rate of 25 fps. The pixels are in the YCbCr format. Using the PixelStreams library of Agility's DK Design Suite, we split the input video signal into two identical streams (see Fig. IV.8). The first stream is fed to the display block and it is converted into the RGB format to display the results on a VGA display. The second stream is fed to the analysis block and it is converted into the grayscale format to reduce (by one-third) the amount of data to be processed.

We can choose to perform the conversion into the RGB format before splitting the input signal and then convert the second stream for the analysis block into the grayscale format. However, this method is not preferable because the conversion from the YCbCr format to the RGB format is approximate. Moreover, in the conversion from the YCbCr format to the grayscale format, the brightness is simply represented by the Y component of the YCbCr format. Further, it is not preferable to approximate the input stream that is fed to the analysis block.

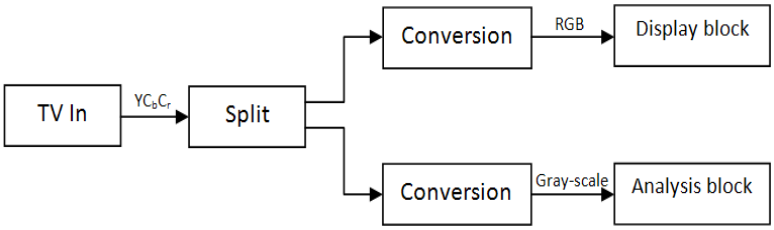


Fig.IV.8. Acquisition block

3.2 Analysis Block

The analysis block consists of several stages. In the first stage, we use inter-image subtraction (delta frames) and apply thresholding to detect moving regions.

To obtain the delta frames, we start by splitting the video signal in three channels (see Fig.IV.9). The first and second channels are used to save the acquired image, creating a delay cell. The image *I(t-1)* is recorded in the memory. The third channel is used to acquire the actual frame at moment *t*. Then, the two image streams are synchronized and fed to the subtraction block. The subtraction block is a modified block that takes the absolute result of subtraction and compares it to a threshold. This function is realized using a macro. The threshold value *Th* is fixed according to the luminosity of the scene.

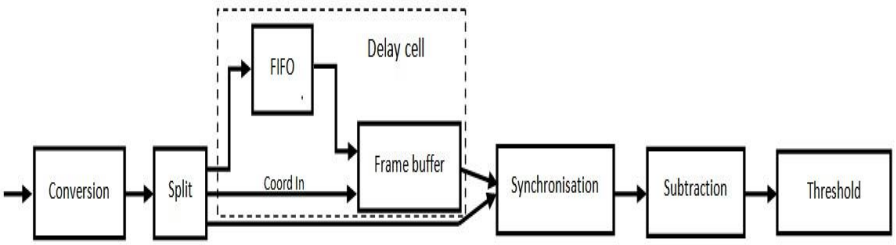


Fig.IV.9. Motion detection block

The second stage of the analysis block involves statistical analysis. In this stage, we search for the min and max values along the x- and y-axes of the mobile regions (Fig. IV.10). In general, this stage must be preceded by a filter for noise reduction. We employed a morphological filter (e.g., alternating sequential filter, opening/closing filter) using the PixelStreams library.

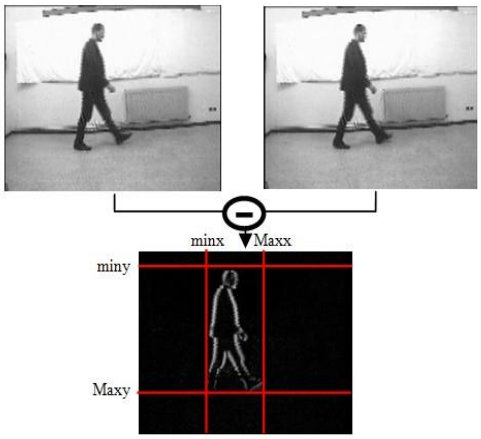


Fig.IV.10. Inter-image difference and calculation of min and max values

After calculating the min and max values along the two axes, we determine the center of gravity of the detected object. We calculate the sum of the pixel co-ordinates that have non-zero values along the x- and y-axes, and we divide these coordinate values by their sum. However, for our implementation, it is better to avoid this division. Therefore, we use the direct method; we subtract the max from the min and divide the result by 2. Division by 2 is achieved by a simple bit shift (right shift). Once the values minX, MaxX, minY, MaxY, and X_G, Y_G are obtained, we copy these values into the classification block and the behavior change detection block. Then, we reset these values to zero.

In the classification block, we calculate the Width and the Height for each moving objects, using Min and Max values along the X and Y axis. Also to avoid the division we

have replaced the condition to detect human beings $(Height/Width) > 1.1$ by the condition $(10.Height > 11Width)$, the same for detecting vehicles the condition becomes $(10.Height > 7Width)$.

3.3 Behavior Change Detection Block

As stated in the previous section, the analysis block provides the behavior change detection block with the parameters of the moving objects. In this stage, we save the values extracted from the first delta frame ($x_g(t-1)$, $y_g(t-1)$, *minx(t-1), MaxX(t-1), miny(t-1), MaxY(t-1)),* and from the second delta frame, we obtain the current values $x_g(t)$, $y_g(t)$, *minx(t), MaxX(t), miny(t),* and *MaxY(t)*. From these latter results, we can calculate the width and length of the moving object to classify the object as human, vehicle, or others. Using the values extracted in two different instants (*t-1, t),* we define the changes in behavior.

Velocity change detection

For velocity change detection, the speed and acceleration are calculated using the two equations presented in Sec. 4.4.1.

However, we simplify these equations by calculating the absolute differences between two moments (the previous and current values). If the absolute difference exceeds a certain threshold V_{th}, we assume that the velocity has changed, and we copy the values of the center of gravity in the display block in order to draw a rectangle around the object. Then, the current values are saved as previous values for the next frame.

Considering a practical problem that involves the values of the center of gravity, in our algorithm, we need to reset all the variables to zero. Consequently, the coordinates of the center of gravity will be zero. If an object enters the scene, the coordinates of the center of gravity change from 0 to X_g, Y_g, and this will cause false detection, Fig. IV.11.

To overcome this problem, we have to ensure that the object has entered the scene entirely. For this purpose, we set a condition on the coordinates of the bounding box for two consecutive instants; if this condition is met ($|minX_{t1} - minX_{t2}| > S$ AND $|MaxX_{t1} - MaxX_{t2}| > S$), we can guarantee that the object has entered the scene entirely, either from the right or from the left, Table IV.1.

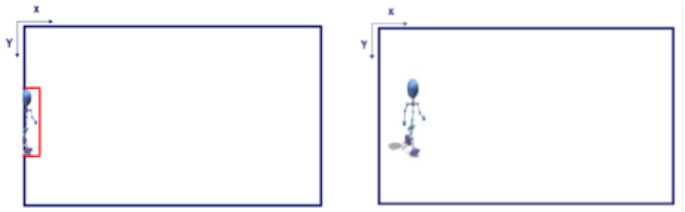


Fig.IV.11. False velocity change detection (the object enters the scene)

Table IV.1 Solution proposed for false velocity change detection

$C_1 \Leftrightarrow \lvert \min x(t-1) - \min x(t) \rvert > S$	$C_2 \Leftrightarrow \lvert MaxX(t-1) - MaxX(t) \rvert > S$	$(C_1 \,\&\&\, C_2)$
0	0	0
0	1	0
1	0	0
1	1	1

Figure IV.12 shows the different stages of this implementation.

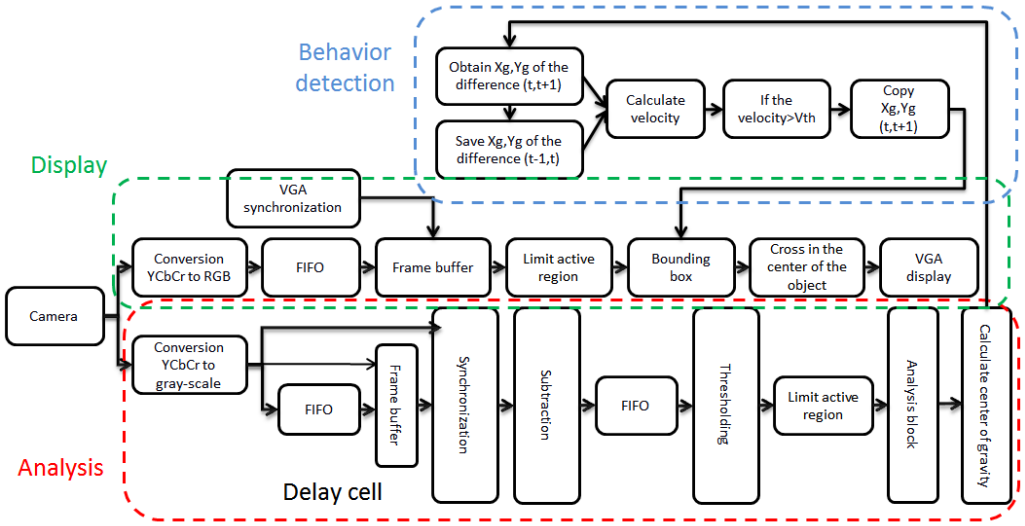


Fig.IV.12. Hardware architecture for velocity change detection

Direction change detection

To implement this application, we follow the same stages as those used in velocity change detection, except that the condition changes. We use the same parameters, *minX*, *MaxX*, *minY*, and *MaxY*, in order to avoid the center of gravity problem. We calculate the difference between *minX1* and *minX2*, and *MaxX1* and *MaxX2*. If there is a change in sign, we assume that the object has changed its direction.

We can easily determine the direction of motion of an object by applying the same concept as that described above. However, in this case, it is impractical to compare the differences between the previous values and the current values to zero because the presence

of a small or non-significant movement (such as that of the arms) can cause false detection. Therefore, to overcome this problem, we compare the difference with a threshold Th_d, which should not be very large. Then, the values *minX*, *MaxX*, *minY*, and *MaxY* will be copied to the block that draws the bounding box.

We can use two blocks for detection in two directions (a different color for each direction of motion). In order to minimize resource consumption, we used only one block for drawing the bounding box by changing the parameters of entry in our macro. In this macro, we added a parameter that changes the color according to the direction of detected motion (Fig.IV.13).

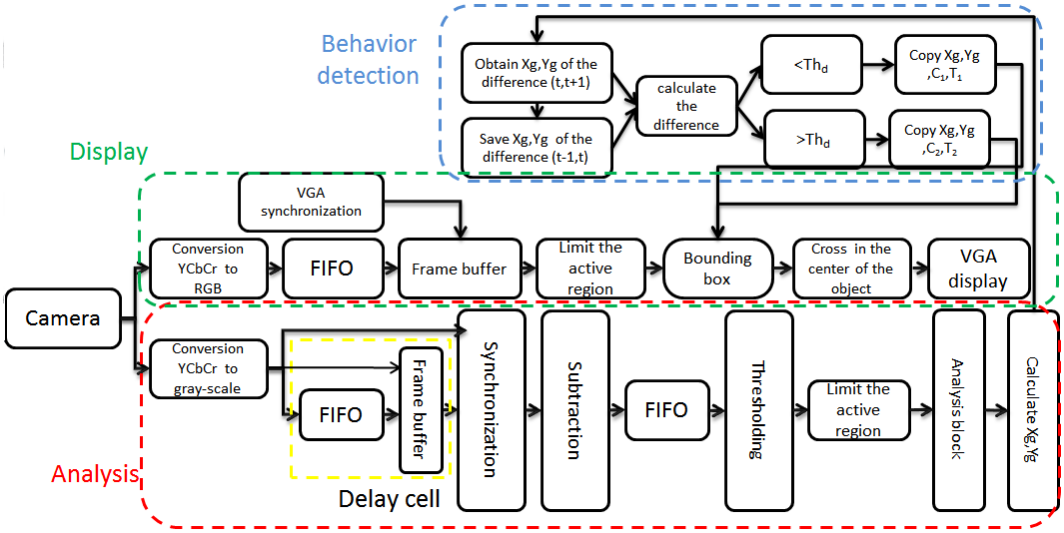


Fig.IV.13. Hardware architecture for direction change detection

Posture change detection

We are interested in such an application to detect a person who leans (bends) to place or pick up something, especially in sensitive locations (e.g., subways). In this case, we are interested in movements along the y-axis of the image (up/down motion), and we use the same architecture as that used in velocity change detection. We calculate the difference between the previous and current values of *miny(t-1), MaxY(t-1), miny(t), MaxY(t)*.

If the difference between the previous and current values is positive, we assume that the person leans, and we copy the values *minX*, *MaxX*, *minY*, and *MaxY* to the block that draws the bounding box and fix the color parameter of the rectangle. We can add a warning message using the PxsConsole filter of PixelStreams. In the opposite case, we assume that the person rises, and we copy the values to the block that draws the rectangle, which uses a different color in this case. As in the case of direction change detection, it is better to use a

threshold Th_p to reduce the occurrence of false detection due to small movements along the y-axis. For such detections, we require a camera whose front sight faces the scene.

Motion analysis

Here, we tried to collect all the above-mentioned behaviors using a single program in order to practically validate the system. To minimize resource consumption, we considered our problem as a finite state machine with several scenarios. The thresholds of detection for each case were used to define and manage these various scenarios. The differences between the values of *minX*, *MaxX*, *minY*, and *MaxY* at moments *t* and *t-1* are denoted by Δ*minx*, Δ*MaxX*, Δ*miny*, and Δ*MaxY*, respectively.

In the first state, all the values are initialized (State 0); they represent the initial state of each new inter-image difference. In the second state (State 1), if the ***absolute*** values of Δ*minx and* Δ*MaxX* are higher than V_{Th}, we assume that the velocity changes and we return to the initial state after copying the values of the block to the bounding box filter. In the opposite case, we go to the third state (State 2) and compare Δ*minx and* Δ*MaxX* with the threshold Th_d. According to the result of this comparison, we assume that a leftward or rightward movement has occurred. Then, we return to the initial state. Starting from this state, if the moving object accelerates, we return to the second state of velocity change. For posture change detection, the condition is related to the values of Δ*miny* and Δ*MaxY* (State 3). We can detect this behavior from any state (e.g., a person runs and leans to collect something). The following figure summarizes these states and the possible scenarios.

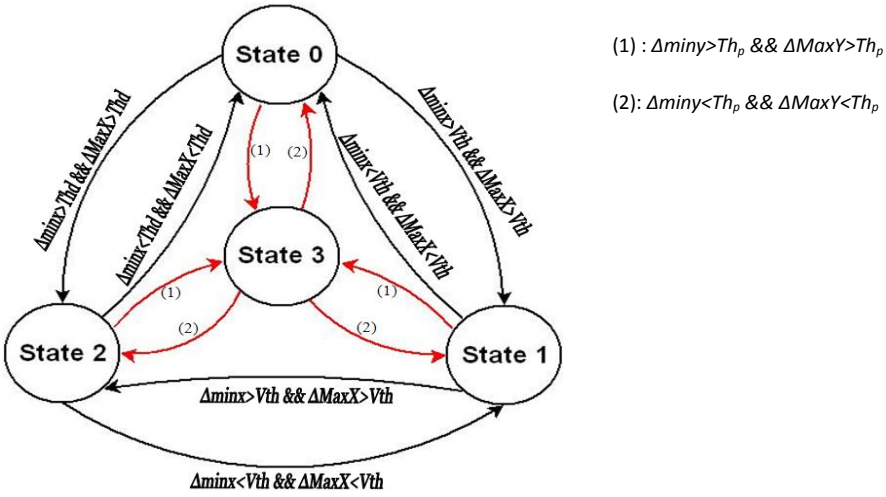


Fig.IV.14. FSM of motion analysis

3.4 Display Block

In this block, we call the macro PxsAnalyseAwaitUpdate, which allows us to pause the display until an update occurs in the analysis block. We obtain the values *minX*, *MaxX*, *minY*, and *MaxY*. If there is a motion, we copy these values to the bounding box filter to draw the rectangle. The values of the center of gravity, X_g,Y_g, are also copied to the PxsCursor filter in order to draw a cross at the center of the moving object. We can add a warning message, e.g., "Warning: velocity change detection", by using the PxsConsole filter of the PixelStreams library. Finally, the results are displayed in the RGB format on a VGA display.

4. HUMAN MACHINE INTERFACE

To make our implementation more flexible, we used a humane machine interface HMI. This interface simplifies not only the use of the hardware but also the change between soft data and hard data, especially for image processing applications that need many parameters to be changed. For example, the parameters of convolution filters and threshold levels. In our implementation, we use Handel-C for the hardware part.

The software part is developed using Visual C++. After generating the bit file using Agility's DK Design Suite [27], we use our software interface to load this bit file via the parallel port (with a frequency of 50 MHz) on the RC200E board in order to configure the FPGA. The algorithm parameters are transferred through this port as 8-bit data at the same frequency. For the user, these operations are hidden. The graphical user interface allows the user to configure/erase the FPGA and change the algorithm parameters. For example, in our case, we can change the threshold level according to the brightness of the scene or the velocity level according to the object in motion (human, vehicle).

5. EXPERIMENTAL RESULTS

An RC200E board with an embedded Virtex-II XC2V1000 FPGA was used for our implementation. The language used was Handel-C. The results for classification and recognition of each behavior are summarized in Tables IV.2–6.

These tables specify the resource consumption and maximal frequency of each implemented detection case for PAL video with a resolution of 720×576.

Table IV.2 Resource consumption and maximum frequency for object classification

Resources	Total	Two objects
I/O	324	179 (55%)
LUTs	10240	2074(20%)
Slice Flip/Flops	10240	2817(27%)
CLB slices	5120	2844(55%)
Block RAM	40	9(22%)
Frequency	/	70,55MHz 5,88ms/image

Table IV.3 Resource consumption and maximum frequency of implementation for velocity change detection

Resources	Total	One object	Two objects
I/O	324	179 (55%)	179 (55%)
LUTs	10240	2286(22%)	3484(34%)
Slice Flip/Flops	10240	3046(29%)	3738(36%)
CLB slices	5120	3092(60%)	4040(78%)
Block RAM	40	9(22%)	9(22%)
Frequency	/	67,21MHz 6,17ms/image	56,85MHz 7,29ms/image

Table IV.4 Resource consumption and maximum frequency of implementation for direction change detection

Resources	Total	One object	Two objects
I/O	324	179 (55%)	179 (55%)
LUTs	10240	2052(20%)	2991(29%)
Slice Flip/Flops	10240	2908(28%)	3491(34%)
CLB slices	5120	2895(56%)	3698(72%)
Block RAM	40	9(22%)	9(22%)
Frequency	/	66,69 MHz 6,22ms/image	55,12MHz 7,26ms/image

Table IV.5 Resource consumption and maximum frequency of implementation for up/down motion detection

Resources	Total	One object	Two objects
I/O	324	179 (55%)	179 (55%)
LUTs	10240	2100(20%)	3233(31%)
Slice Flip/Flops	10240	2927(28%)	3595(35%)
CLB slices	5120	2961(57%)	3904(76%)
Block RAM	40	9(22%)	9(22%)
Frequency	/	67,14MHz 6,17ms/image	58,97MHz 7,03ms/image

Table IV.6 Resource consumption and maximum frequency of implementation for motion analysis

Resources	Total	Two objects
I/O	324	179 (55%)
LUTs	10240	3640(35%)
Slice Flip/Flops	10240	4173(40%)
CLB slices	5120	4537(88%)
Block RAM	40	9(22%)
Frequency	/	53,18MHz 7,80ms/image

In all these implementations, the results show that the two main constraints, i.e., the resource limit of our FPGA and the real-time aspect (40 ms/image), are well respected. We note that the consumption of the CLB blocks increases in the case of detection of multiple objects; this is caused by the algorithm used to identify the number of objects in the scene. We also note that the algorithm for motion analysis that collects all the previous behaviors can be implemented on our FPGA in real time, but it consumes nearly all of the CLB resources (88%).

The following figures show the results of all these implementations. Figure 15 shows the results of moving object identification.

Fig.IV.15 Classification of detected objects

For action recognition, each behavior is represented by a different color, and a warning message is added below the scenes.

Figure.IV.16 shows the results of velocity change detection in the case of one object. In Fig. IV.16.a, we note that as soon as the object decreases its speed, the rectangle disappears. In Fig. IV.16.b, as soon as the object starts to run, a rectangle appears around it.

(a)

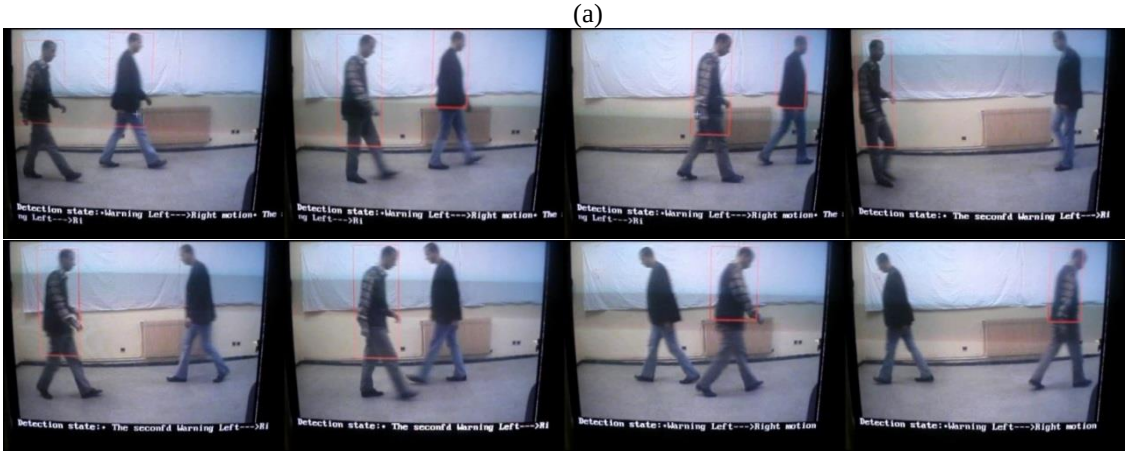
(b)

Fig.IV.16. Results of velocity change detection in the case of one object

Figure.IV.17 shows the results of velocity change detection in the case of two objects. As soon as the objects start running, a rectangle appears. We note that in the case of occlusion, the algorithm considers both objects as a single object. After the objects separate, two rectangles with different colors appear on them.

Fig.IV.17. Results of velocity change detection in the case of two objects

(a)



(b)

Fig.IV.18. Direction change detection

Figure.IV.18 shows the results of direction change detection. Fig. IV.18.a shows direction detection for two directions: right to left movement, represented by the blue rectangle, and

left to right movement, represented by the red rectangle. The figures also show warning messages below the images. Fig. IV.18.b shows the results of direction change detection in one direction for two objects.

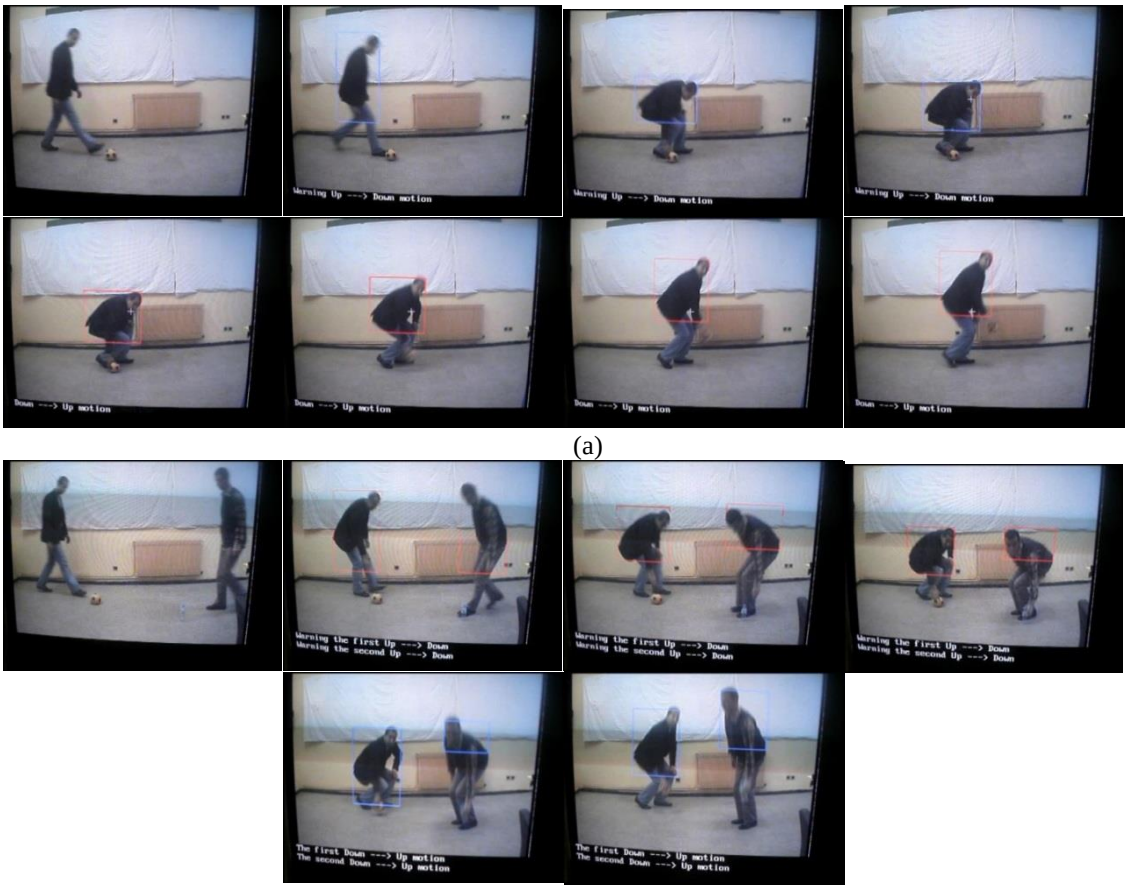


Fig.IV.19. Posture change detection: a) for one object, b) for two objects

Figure IV.19 shows the results of posture change detection. When the object leans to pick up something, it will be detected. Up/down and down/up motion are represented in different colors. A warning message is added in each case.

Figure IV.20 shows the results of collecting all the behaviors using a single program. Motion to the right and left are represented by red and blue rectangles, respectively. Further, up/down and down/up motion are represented by turquoise and yellow rectangles, respectively. Finally, velocity change is represented by a black rectangle. In every case, a warning message is displayed.

Fig.IV.20. Motion analysis

Figure IV.21 shows our human machine interface (HMI), which is divided into four sections. Three of these sections are used to detect just one simple behavior each, whereas the fourth section detects all the behaviors. Using this HMI, we can send the bit-file for configuring or erasing our FPGA, or directly changing the filter parameters without the need to use the IDE.

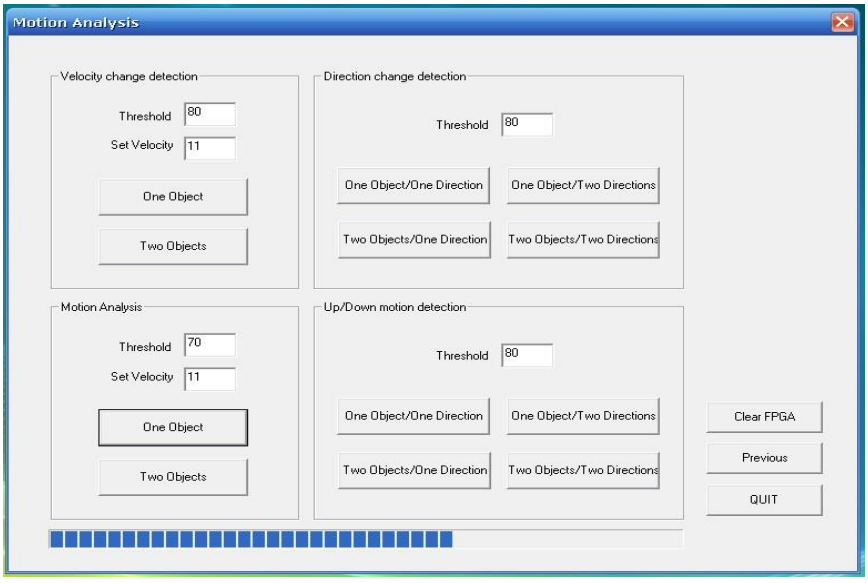


Fig.IV.21. Humane machine interface

6. CONCLUSION

In this work, we presented a human machine interface that simplifies the use of the hardware part by enabling us to communicate with it using the graphical user interface. In addition, it simplifies the choice of the algorithm to be implemented and modifies the parameters of this algorithm. We adopted the proposed approach for object detection and behavior recognition based on motion analysis and sudden movements. We exploited the hardware part, which offers the possibility of handling large amounts of data and performing calculations for image processing via parallel processing, guaranteed by the use of the PixelStreams library of Agility's DK Design Suite. Further, we tried to improve our architecture by collecting all the different behaviors using a single program. In addition, we added warning messages using the PxsConsole filter. Thus, we successfully implemented different algorithms that can recognize objects in motion and detect changes in velocity, direction, and posture in real time. The results showed that our approach achieves good recognition and detection of these behaviors, especially in indoor areas. However, in outdoor areas, the results are less promising owing to the simple motion detection algorithm used. This problem is aggravated by occlusion due to overlapping movements of different persons. Therefore, in the future, we will try to use multiple cameras (stereoscopic, Kinect) with improved motion detection and learning methods to detect behavior changes in crowded environments.

In the next chapter, we will present another intelligent video surveillance system that can detect falls. The system uses a new technique to extract features and machine learning methods to enhance accuracy.

REFERENCES

[1] L.Wang, W. Hu, and T. Tan, "Recent developments in human motion analysis", Pattern Recogn, 36 (3), 585-601, (2003).

[2] W. Hu, T. Tan, L. Wang, and S. Maybank, "A survey on visual surveillance of object motion and behaviors", IEEE T Syst Man Cyb, 34 (3), (2004).

[3] T. Ko, "A survey on behavior analysis in video surveillance for homeland security applications", AIPR 2008, Washington, DC, USA, 2008.

[4] M. Piccardi "Background subtraction techniques: a review", IEEE SMC, 4, 3099-3104, (2004).

[5] G.G.S. Menezes and A.G. Silva-Filho, “Motion detection of vehicles based on FPGA”, SPL VI Southern, 151-154, (2010).

[6] W.Shuigen, C.Zhen, L.Ming, and Z.Liang, “An improved method of motion detection based on temporal difference”, ISA 2009, 1-4, (2009).

[7] Widyawan, M.I. Zul, and L.E. Nugroho, “Adaptive motion detection algorithm using frame differences and dynamic template matching method”, URAI 2012, 236-239, (2012).

[8] I. Ishii, T. Taniguchi, K. Yamamoto, and T. Takaki, “1000 fps real-time optical flow detection system”, Proc. SPIE 7538, 75380M (2010).

[9] J. Diaz, E. Ros, F. Pelayo, E.M. Ortigosa, and S. Mota, “FPGA-based real-time optical-flow system”, IEEE T Circ Syst Vid, 16, (2), 274-279, (2006).

[10] M. Paul, S. Haque, and S. Chakraborty, “Human detection in surveillance videos and its applications - a review”, EURASIP JASP, Springer International Publishing, (2013).

[11] Z. Chen, T. Ellis, and S. A. Velastin, "Vision-based traffic surveys in urban environments," Journal of Electronic Imaging, vol. 25, pp. 051206-051206, 2016.

[12] M. Elhamod and M. D. Levine, "Automated real-time detection of potentially suspicious behavior in public transport areas," IEEE Transactions on Intelligent Transportation Systems, vol. 14, pp. 688-699, 2013.

[13] C. Li, P. Wang, S. Wang, Y. Hou, and W. Li, "Skeleton-based action recognition using LSTM and CNN," in Multimedia & Expo Workshops (ICMEW), 2017 IEEE International Conference on, 2017, pp. 585-590.

[14] H. Su, H. Yang, S. Zheng, Y. Fan, and S. Wei, "The large-scale crowd behavior perception based on spatio-temporal viscous fluid field," IEEE Transactions on Information Forensics and Security, vol. 8, pp. 1575-1589, 2013.

[15] L. Onofri, P. Soda, M. Pechenizkiy, and G. Iannello, "A survey on using domain and contextual knowledge for human activity recognition in video streams," Expert Systems with Applications, vol. 63, pp. 97-111, 2016.

[16] F. Cupillard , A. Avanzi , F. Bremond, and M. Thonnat, “Video understanding for metro surveillance”, ICNSC 2004.

[17] I. Haritaoglu, D. Harwood, and L. S. Davis, “W4: Real-time surveillance of people and their activities”, IEEE T Pattern Anal, 22 (8), 809-830 (2000).

[18] J. Hiraiwa, E. Vargas, and S. Toral, "An fpga based embedded vision system for real-time motion segmentation," in Proceedings of 17th International Conference on Systems, Signals and Image Processing. Brazil, 2010.

[19] Z. Hou, H. Zhu, N. Zheng, and T. Shibata, "A single-chip 600-fps real-time action recognition system employing a hardware friendly algorithm," in Circuits and Systems (ISCAS), 2014 IEEE International Symposium on, 2014, pp. 762-765.

[20] A. Al Maashri, M. Debole, M. Cotter, N. Chandramoorthy, Y. Xiao, V. Narayanan, et al., "Accelerating neuromorphic vision algorithms for recognition," in Design Automation Conference (DAC), 2012 49th ACM/EDAC/IEEE, 2012, pp. 579-584.

[21] M. Komorkiewicz and M. Gorgon, "Foreground object features extraction with GLCM texture descriptor in FPGA," in Design and Architectures for Signal and Image Processing (DASIP), 2013 Conference on, 2013, pp. 157-164.

[22] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in Advances in neural information processing systems, 2012, pp. 1097-1105.

[23] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2014, pp. 580-587.

[24] “Virtex II 1.5v Field-Programmable Gate Arrays”, Data sheet, Xilinx Corporation, 2001.

[25] “DK5 Handel-C language reference manual”, Agility 2007.

[26] “PixelStreams Manual”, Mentor Graphics Agility (2012), http://www.mentor.com/products/fpga/handel-c/pixelstreams/

[27] “Agility DK User Manual”, Mentor Graphics Agility (2012), http://www.mentor.com/products/fpga/handel-c/dk-design-suite/

Chapter V

Fall Detection System Based on Combination of Features and Motion Analysis

Don’t throw away the needle when you get a sword!

1. INTRODUCTION

Most elders nowadays live alone at their houses, nursing homes and sometimes in private room which could cause many risks, and that can be dangerous to their physical health. This physical harm can be caused by different situations. For example, accidental falls, medical problems (loss of consciousness, tonic-clonic seizures) or even assaults of aggression. Some elders live with their families that take care of them, but the danger is still possible for example when falling in glistening floors, hitting sharp objects or from bed when sleeping. That can lead to several injuries: skull and bone fractures, contusions, concussions and deep flesh wounds. When these cases happen, the injured person would need immediate rescue, or the possibility of death would be high. For that, many studies have been done over the last decade to develop intelligent systems that can prevent and detect when the fall occurs [1-4]. We can classify these systems into five categories according to the technology they use. First, vibration/pressure-based systems [5-7]. These systems use sensors placed on the floor in order to analyze the movement of the person and the surface of contact with the floor. Despite of its robustness, the cost of applying such systems is very expensive and depends on many factors, such as the type of floor and the surface. Furthermore, moving big objects like a sofa may generate false alarms. The second category is the accelerometer-based systems. They are presented as wearable devices [8-10], combined with gyroscopes and actimetry sensors seem to be the ideal solution for detecting fall occurrences. However, as wearable devices, they must be worn all day which can make elder people feel uncomfortable with it, and sometimes forget to put it on. Additionally, we should note the autonomy problem of these smart devices. For the third category, we find the radar-based systems which use back-scattered waves 'Doppler effects' [11-13] to track daily activities. These systems are in general composed of different connected or separated sub-systems and present many advantages such as insensitivity to illumination change and privacy preservation for the tracked person. Though, being dependent on radar signals, this technology suffers from numerous false alarms, due to misdetection of falls with other human movements like laying or sitting. Also animal movements, and interior walls and objects cause clutters and ghost targets due to scattered signals [12]. The forth category which is Electroencephalography-based systems (EEG) [14-16] have recently been developed to analyze brain signals when falls occur. However, this technology is still new. Therefore, we cannot confirm its efficiency yet. Furthermore, the EEG sensors are expensive and not easy to carry due to its size. The last category is vision-based systems, which has gained huge importance in the last decade because of many reasons: it

does not require to be worn, it does not need special installation, it can cover great surfaces, it can uses different camera sensors (RGB, thermal, depth camera,…) to enhance accuracy and solve privacy problems. Different approaches (shape analysis, gait analysis, motion analysis, deep learning…) developed in order to set an accurate fall detection system, and now presented below.

2. RELATED WORK

Many vision-based systems for fall detection have been developed in this last decade. For example, Lee *et.al*[17] developed an intelligent system for fall detection using a digital camera installed in the ceiling. They used a shape feature vector composed of five elements to represent the silhouette of the person (center of gravity, perimeter of the object, the ferret diameter, and velocity of the center of gravity). This system can detect a fall in 77% of cases and has false alarms rat of 5%. This is due to system limitations such as the use of simple background subtraction method which may heavily alter the results, and the use of few shape features to represent a fall.

Rougier *et.al* [18] designed a video surveillance system for fall detection based on human shape deformation using mean matching cost and Procrustes distance. The tests were applied on their own data-set which consists of different viewpoints captured using eight IP cameras. A Gaussian mixture model was used to classify different activities into fall or not. The authors report that best classification error rate for Procrustes distance for each camera is less than 10% and 2,7% using majority vote (at least three of four cameras). For matching cost their system gave more than 98% with majority vote. However, being based on two features, this method suffers from high false alarm in case of occluding object and small movements. Even in the case of majority vote using multiple camera (which is a very expensive solution, cost and real time data processing) the problem of small movements still persist. For example, a person walking stops and opens the fridge, in this case just his arms or his high part of the body is detected which will generate high cost matching and Procrustes value between the two instants, and unfortunately this case is not handled in the used dataset.

Another study [19] used the simple deep learning network PCANet and SVM for detection and classification of fall detection. Experiments was conducted on the Multiple Camera Fall dataset [18] and achieved 89,2% sensitivity and 90,3% specificity. Despite the good result obtained the use of deep learning still posing a serious problem for real time application. Moreover, the data-set used consist of 24 scenes with 22 falls conducted by one actor which

are not sufficient for deep neural training. Other similar works that use neural networks can be found in [20,21]. Other techniques try to analyze the gait using video cameras to detect if a fall happened or not [22,23]. In general, these techniques analyze the gait speed by detecting if there is a shuffling or unstable gait to prevent and detect a fall. However, not all falls happened when walking, may a fall happen when trying to stand-up or fall from his/her bed. For that, these systems suffer from a high miss rate. Other works tried to use different cameras sensors as in [24] in which the authors used the IRISYS thermal-imaging sensors consisting of low-element-count infrared array that can locate, track and compute the size and the velocity of thermal target. But due to the low resolution of this system, it cannot detect a third of the fall scenarios. Another work [25] used infrared lights and cameras to monitor nighttime actions of elderly in their homes, first the acquired images were smoothed then the silhouettes was extracted and normalized. A linear SVM classifier was used for action recognition. The results showed that a fall was detected in 83.28%. In [26], the authors used a similar system. Four infrared lights were mounted in the four corners of the room with wide angle camera. Instead of analyzing the silhouette alone the authors analyzed the shadows of the silhouette also to recognize different actions and to increase accuracy. The authors reported that the use of shadows has increased the rate of accuracy to 94,22%. Several other studies [27-29] used depth cameras such as Kinect, taking the advantage of the built-in IR sensor. Experiments showed promising results. However, these infrared systems needs a special study to measure the rate of different materials absorption of IR light, in addition, IR light flashing may be uncomfortable for elder people. For Kinect systems, the field of view and the minimum and maximum range pose another significant problem. Furthermore, studies [30] report that the Kinect does not necessarily obtain precise body segment lengths during motion, which may affect the result of detection. In [31] the authors focused on detecting/tracking head and computing the vertical velocity of the head as feature for detecting fall. Despite the high accuracy of fall detection, these systems depend on how to detect the head position to initialize the system which depends strongly on the used motion detection method. Further, the use of skin color model to locate head position as in [32] can heavily decrease the accuracy due to different situations such as when the person is not facing the camera, a person is wearing a hat or in the case of poor light conditions.

In this chapter, we propose a fall detection system based on combining shape features and motion analysis in order to increase the accuracy. The main contribution of this work is the development of a new method for computing the vertical velocity of the head without the need

of tracking filter which represent great constraints in time and resources especially for real time application. This method also resolves the dependency to the skin or hair color to estimate head coordinates, which often limits the capability of such systems. Further, previous researches showed that shape analysis gives good results but not in all the cases of daily life activities (ADL). Hence, we combined these features in order to enhance fall detection and reduce false alarms.

3. OUR APPROACH

The system proposed in this work consists of several stages. In the first stage a background subtraction method is applied on each frame in order extract silhouettes of the moving person. The second stage is the post-processing, in which we applied a morphological filter followed with a connected component labeling method. These improvements were done to guarantee that just the silhouette of the person is fed to the next stage. In the third stage, we extract the features of the silhouette (vertical velocity of the head, area, height/width ratio, feret diameter, orientation, Procrustes distance), then a fall test is conducted. The alarm will not be signaled until an inactivity test was done. Fig.V.1 shows general outline of the fall detection system.

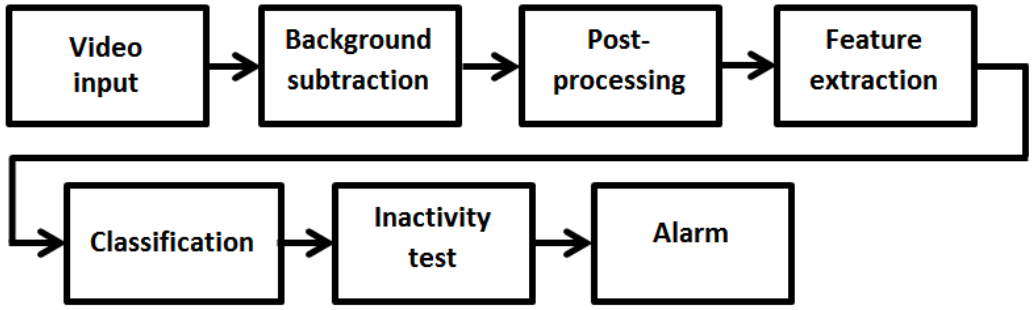


Fig.V.1. General outline of the fall detection system

3.1 Background subtraction

From the studies cited above, we deduced that foreground segmentation and silhouette detection is very crucial for achieving a good accuracy. For that, a research [33] was conducted to choose the appropriate method for background subtraction. Based on this research we selected the simplified self-organized background subtraction method (Simp-SOBS) presented in [34]. This choice is justified for several reasons: good results for extracting the entire shape and its robustness to shadows, low computational time and low memory requirement. These make it suitable for implementation on real time systems. In this method

the background is modeled by a SOM-like neuron map. In which each pixel $I_t(x, y)$ is represented in the HSV color space. Hence, each neuron $b(x, y)$ has three inputs h_b, s_b, υ_b. After initialization, a test match is done using:

$$d\left(b, I_t(x, y)\right) \le Th \quad \wedge \quad \left|\upsilon_{I_t} - \upsilon_b\right| \le Th_v \tag{V.1}$$

where $d\left(b, I_t(x, y)\right)$ is the Euclidean distance in HSV color space, defined by:

$$d\left(b_i, I_t(x, y)\right) = \sqrt{\begin{aligned}&\left(\upsilon_{b_i} s_{b_i} \cos\left(h_{b_i}\right) - \upsilon_{I_t} s_{I_t} \cos\left(h_{I_t}\right)\right)^2 + \\ &\left(\upsilon_{b_i} s_{b_i} \sin\left(h_{b_i}\right) - \upsilon_{I_t} s_{I_t} \sin\left(h_{I_t}\right)\right)^2 + \left(\upsilon_{b_i} - \upsilon_{I_t}\right)^2\end{aligned}} \tag{V.2}$$

Th and Th_v are threshold values, automatically computed using Otsu's threshold method. A minimum value was set to 13 to overcome the problem of sparse matrix. If there is a match, the pixel is considered as background and the weights of the corresponding neuron $b(x, y)$ and its neighbors are updated using:

$$b_{t+1}(x, y) = b_t(x, y) + \alpha_1\left(I_t(x, y) - b_t(x, y)\right) \tag{V.3}$$

$$b_{t+1}(x', y') = b_t(x', y') + \alpha_2\left(I_t(x', y') - b_t(x', y')\right) \tag{V.4}$$

where $x' = x - 1, x + 1$ and $y' = y - 1, y + 1$ are the coordinates of the neighboring neurons, and α_1 and α_2 are the learning rates, with $\alpha_1 > \alpha_2$ for non-uniform learning.

If there is no match, $I_t(x, y)$ is considered as foreground pixel and no update is required.

3.2 Post-processing

In this stage, after morphological filtering, a simple connected component (CC) labeling algorithm is used to extract regions. 8-pixel neighborhood connectivity is employed. For each foreground pixel, we search for unlabeled pixel then we apply a flood-fill algorithm to label all pixel neighbors. We repeat these steps until all the pixels are labeled. In order to exclude any moving objects or pets, we compute the area of each region. Then, all CC that satisfying Eq. (V.5) will be considered as background:

$$area(i) < \frac{area_{\max}}{2} \tag{V.5}$$

This stems from the fact that the person, in his/her room, has the biggest area when moving or falling compared to pets or objects. Also, in the case of slow motion (e.g. arm movements) all moving objects will be considered as background since we are interested on fall detection not on motion.

3.3 Feature extraction

In order to determine that a fall is happened or not, we have to extract features of the silhouette. Each silhouette is represented by a six-dimensional vectors composed of the following features: difference of area, height/width ratio, orientation of the silhouette, ferret diameter, Procrustes distance and vertical velocity of the head.

- Vertical velocity of the head & orientation

To compute vertical velocity of the head, we have to estimate its center of gravity. First, the minimum area oriented box is computed using feret diameter. For each angle from $\theta = 0$ to $\theta = 90^{\circ}$, the area is computed by:

$$area(\theta) = fd(\theta) \times \mathrm{fd}(\theta + 90^{\circ}) \tag{V.6}$$

By computing the minimum oriented box (Eq.(V.7)), we find also the orientation of this silhouette. The length L and width W correspond to the largest and the smallest feret diameter for this angle respectively.

$$[\mathrm{Minarea}, \Theta] = \min(area(\theta)) \tag{V.7}$$

Using the convex hull vertices of the silhouette, we compute the approximate center (c_x, c_y) of the minimum oriented rectangle (Fig.V.2).

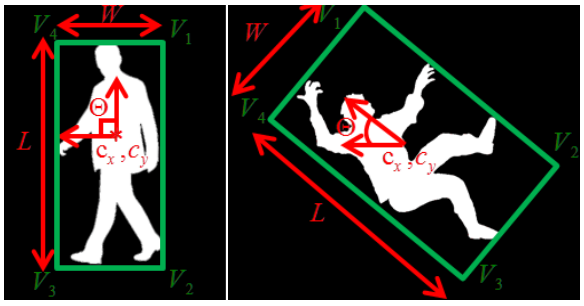


Fig.V.2. Features extracted using feret diameter and minimum of oriented box

Since the convex hull that envelops the silhouette has not always a rectangle shape, we need to compute the vertices of the oriented box (V_1, V_2, V_3, V_4) using the Eq.(V.8).

$$\begin{cases} V_{i_x} = \left(V_{i_x}^{'} - c_x\right)\cos(\theta) - \left(V_{i_y}^{'} - c_y\right)\sin(\theta) + c_x \\ V_{i_y} = \left(V_{i_x}^{'} - c_x\right)\sin(\theta) + \left(V_{i_y}^{'} - c_y\right)\cos(\theta) + c_y \end{cases} \quad (V.8)$$

where $\left(V_{i_x}^{'}, V_{i_y}^{'}\right)$ are the vertices of the horizontally aligned rectangle obtained using its center of gravity $\left(c_x, c_y\right)$ and its length L and width W .

In case of walking Fig.V.3 (a), head center coordinates are computed using Eq.(V.9) where $\theta \approx 90^{\circ}$:

$$\begin{cases} x_h = x_{23} - L_h \cos(\theta) \\ y_h = y_{23} - L_h \sin(\theta) \end{cases} \quad (V.9)$$

x_{23} and y_{23} are computed using the vertices of the oriented rectangle.

$$\begin{cases} x_{23} = \dfrac{V_{2_x} + V_{3_x}}{2} \\ y_{23} = \dfrac{V_{2_y} + V_{3_y}}{2} \end{cases} \quad (V.10)$$

L_h is the height from $\left(x_{23}, y_{23}\right)$ coordinates to the center of gravity of the head. It is computed based on the study of Naini *et al.*[35] in which the authors reported that the ideal craniofacial height (head length H_L) to standing height proportion is in the range of 1/7 to 1/8.5. This was also used in ancient civilization arts (Augustus status and da Vinci's Vitruvian man art). Using this proportion we can estimate the head length $H_L = L/8$. Then, we will compute L_h, Eq.(V.11), Fig.V.3. a.

$$L_h = L - \left(\frac{H_L}{2}\right) \quad (V.11)$$

Eq.(V.9) is applied in case of backward fall (a fall away from the camera). In case of forward fall (a fall toward the camera) we use:

$$\begin{cases} x_h = x_{14} + L_h \cos(\theta) \\ y_h = y_{14} + L_h \sin(\theta) \end{cases} \quad (V.12)$$

Fig.V.3 explains these different scenarios. (b-c) are the case of forward fall and (d) the case of backward fall. We assume that if the person is moving forward/backward he/she will fall forward/backward.

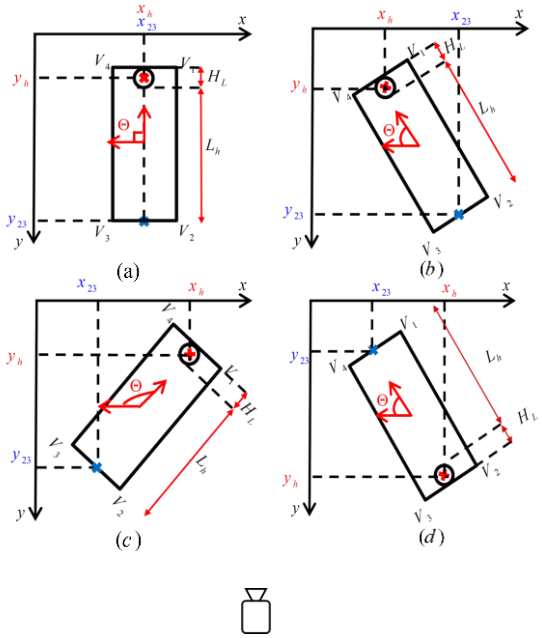


Fig.V.3. Different scenarios of backward/forward fall

Fig.V.4 shows the case of forward fall (a fall toward the camera) and backward fall (a fall away from the camera) on real dataset.

Fig.V .4. Forward/backward fall

For that, we should first detect the direction of the movement when walking. Hence, a pre-test for non-possible fall/bend is done, using angle variation and feret diameter variation with $\theta = 90^{\circ}$ in the two successive instants (a big variation on these two features may represent a fall/bend movements).

In this case, we will detect the direction using the finite state machine (FSM) represented in Fig.V.5, where state (0) represents the initial state, state (1) and state (2) represent the backward/forward movement respectively, state (3) represents the case of horizontal

movement. We note that the last state is a rare condition but not impossible, because when a person falls, it is certainly moving toward or away from the camera which bring us to state (1) or (2), for that reason the state (3) can be eliminated.

The states (S1, S2, S3) are determined by the variation of the peak height of the silhouette (conditions C_1, C_2 and C_3 in eq.V.13).

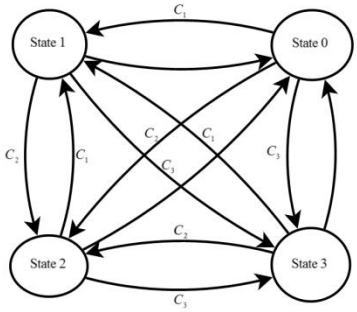


Fig.V. 5. Finite state machine for direction change detection

After that, using equation (V.9) or (V.12) depending on the final state, we compute the estimated center of gravity. In case of possible fall/bend movement, we use the previous state to determine which equation we use.

$$\begin{cases} C_1 = \left(y_{top} - y_{topprev} \right) > 0 \\ C_2 = \left(y_{top} - y_{topprev} \right) < 0 \\ C_3 = \left(y_{top} - y_{topprev} \right) = 0 \end{cases} \quad \text{(V.13)}$$

Lastly, we compute the vertical velocity of the head using its previous and its current center of gravity. Finally, we should note that the fall of small objects will note alter the results owing to the condition set in the pre-processing stage.

- Procrustes distance

Procrustes distance [36] is widely used in medical image analysis and biology. two version exist of this measure: the ordinary Procrustes analysis (OPA) used for measuring shape deformation or similarity, and the generalized Procrustes analysis (GPA) used for obtaining an average shape from two configurations [37,38]. Recently, OPA was used for human action classification [39,40], face recognition [41] and gait recognition [42]. Authors of [18] used this tool to measure the silhouette deformation between two instants to detect fall for elderly

people. Despite the good results obtained in this study, the use of shape analysis alone is not sufficient because it may generate false positives, for instance when a person walks then stops and opens the door of fridge or a person walking he gets hidden (occluded) by furniture. These actions will generate high Procrustes distance between two instants. Fortunately, when combining this measure with other features, we can reach high accuracy system taking into account all challenges presented in ADL.

The two configurations matrices (landmarks) X and Y , used to compute the Procrustes distance, represent the coordinates of N boundary points equally spaced of the silhouette at two successive instants. OPA is obtained by minimizing the squared Euclidean distance taking into account different transformations (translation $a+ib$, rotation θ and scale β)

$$D(x,y)=\left\| y-\left(\beta e^{i\theta}x+(a+ib)1_k\right)\right\| \tag{V.14}$$

The minimization of this distance is obtained when (proof in [36]): $\left(\hat{a}+i\hat{b}\right)=0$, $\hat{\beta}=\left\|x^{*}y\right\|$, $\hat{\theta}=\arg\left(x^{*}y\right)$. The minimum of Eq.(V.14) is given by:

$$D_F=\sqrt{1-x^{*}yy^{*}x} \tag{V.15}$$

- Feret diameter

The feret diameter with angle $\theta=90^{\circ}$ is computed and the difference between two instants is used as feature for detecting a fall. The angle $\theta=90^{\circ}$ (Fig.3.a) is chosen based on the assumption that elder people will note carry big objects and also to avoid false alarms generated by arms movement when choosing $\theta=0^{\circ}$.

- Difference of area

The difference in area of the person increases significantly when falling. This is due to the dispersedness of the body, arms or clothes movements during the fall.

- Height/Width ratio

3.4. Lack of movement

As mentioned in [43], the critical phase of fall is very short. It is about 300-500ms. After that, the person stay relatively inactive (long lie), which may indicate his condition,

unconscious or injured. The detection of this state could be important since a person who can move can easily ask for help. For that, the detection of lack of movement is essential. In general, the silhouettes of the fallen person stay inactive with a static (constant) shape for a while before it disappear due to inactivity and the update process of the BGS technique. We can set many criteria for detecting this phase. For example we can use the Procrustes distance or the feret diameter or the orientation of the silhouette. Despite its potential to confirm a fall and diminish the number of false positives, we did not need to implement this step in our study.

4. DISCUSSION AND RESULTS

4.1. Background subtraction and feature extraction

Three major 2D camera-based datasets for fall detection exist: Multiple camera fall dataset [18], L2ei fall dataset [44] and the high quality fall simulation dataset [45]. In this work, we tested our algorithm on the L2ei dataset for two main reasons, the large number of actors (11 actors) and the large number of video segments (250 scenarios in 4 different places). Each scenario may contain different activities (walking, standing up, forward/backward fall, crouching, siting on chair, other daily life activities). From the study conducted in [31] we set, in all the tests, the values of the learning rates used in Simp-SOBS to $\alpha_1 = 0.02$ and $\alpha_2 = 0.01$. The SOM network needs to be initialized with an empty scene. It is in general the initial image I_0. However, almost all the videos in the used dataset do not contain empty scenes. To solve this problem, the median of all scenes was computed to generate an empty scene. In case of poor results, the moving object was removed from the scene manually.

In all the tests the learning rates used in Simp-SOBS for background subtraction was set to $\alpha_1 = 0.02$ and $\alpha_2 = 0.01$. As mentioned in [40] the critical phase of fall takes about 300ms to 500ms. So in order to detect significant change on shape and to reduce the amount of data to be treated, we have chosen one frames every then frames. this is equivalent to 400ms for L2ei dataset frame rate which is 25images/s. The number of landmarks (contour points) used to compute the Procrustes distance was set to $N = 200$ points.

Figure V.6 (a-f) shows the results of applying our algorithm on sequence 40 in a coffee room. From the graphs of the extracted features we can deduce easily in which frame the fall happened. This fall is characterized by increasing values of area change: Procrustes distance,

vertical velocity of the head, orientation change, ferret diameter change and decrease of the H/W ratio. In the beginning of the scene, the person was sitting and stood up which cause a slight increase in the vertical velocity of the head, the Procrustes distance, and the ferret diameter (Fig.V.6.a). Fig.V.7.a-e show the result of background extraction with head coordinates estimation marked with a red dot. The minimum oriented box and the aligned oriented box are shown with different colors. Fig.V.7.f shows the case of a wrong estimation of the head coordinates. In this case the person was getting up from a seat.

4.2. Classification

In order to test the performance of our algorithm on a large dataset, all the video sequences of the coffee room was tested and annotated to two classes 'fall' and 'no fall'. This results in 16554 features (a matrix of 2759 images × 6 features). Three different classifiers were used to determine the best accuracy. Radial Basis Function Support Vector Machine RBF-SVM [46], K-Nearest Neighbor KNN [47] and Backpropagation Neural Network BPNN [48]. In order to test fairly these classifiers, the same training and test sets were applied on these classifiers. These datasets were randomly chosen. 70% of the data for training and 30% for test. For the BPNN, 5% from the training set was used for validation.

The support vector machine is a discriminative learning algorithm developed by Vapnik [46]. The aim is to determine the optimal separating hyperplane and predict correctly the class of the new data. The equation of hyperplane is given by:

$$\langle w, x \rangle + b = 0 \tag{V.16}$$

where $\{x_k, y_k\}_{k=1}^{N}$ represent the training set of N points, x_k are the features vectors and y_k their categories, $w \in \Re^d$, $\langle w, x \rangle$ is the inner (dot) product and b is real.

For most cases, the separating hyperplane (border) is non-linear and for that the model uses different kernels. Table V.1 shows the most used kernels.

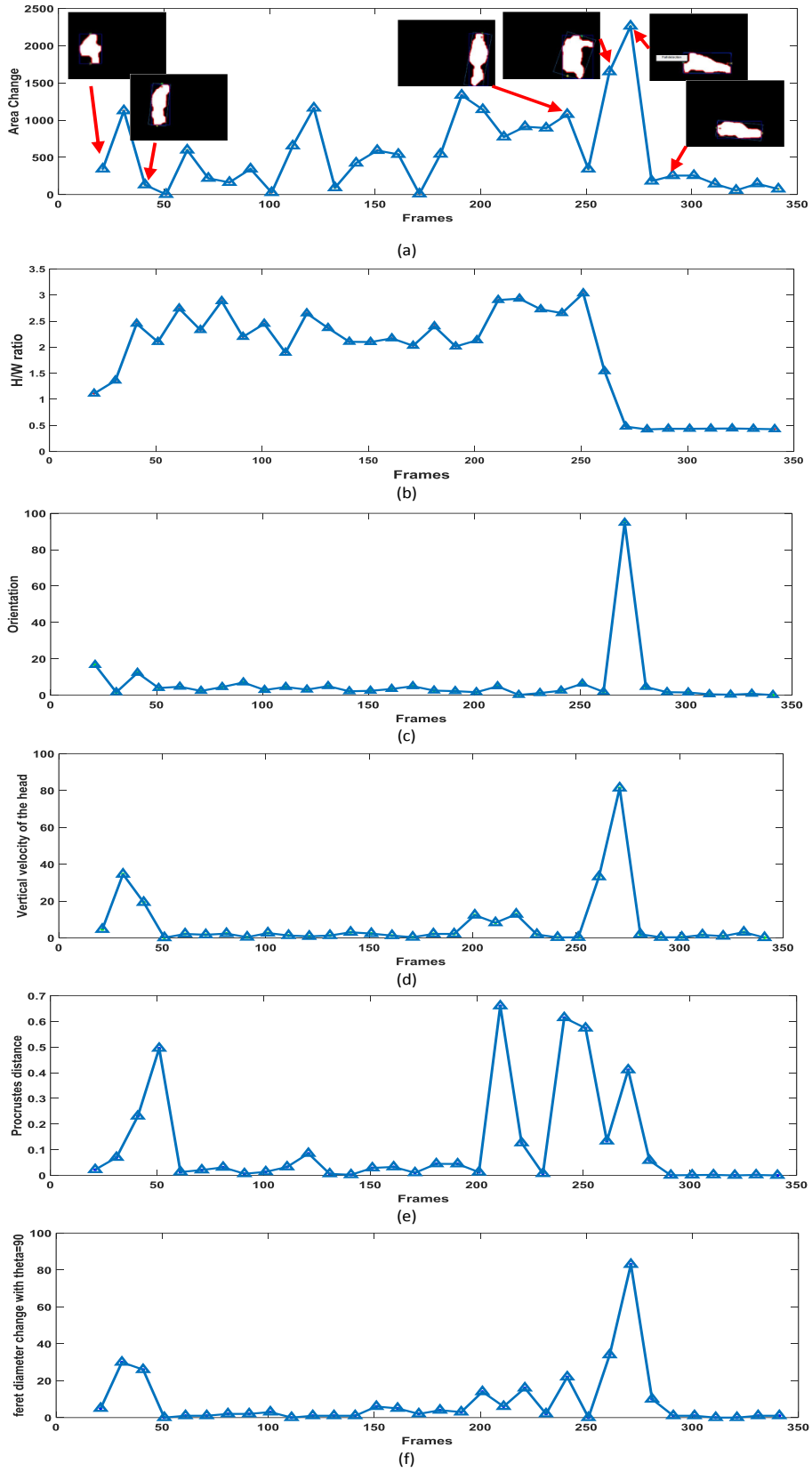


Fig.V. 6. Background subtraction and feature extraction applied on sequence 40: (a) area change with silhouette extraction on some frames, (b) H/W ratio, (c) orientation of the silhouette, (d) vertical velocity of the head, (e) Procrustes distance, (f) ferret diameter with $\theta = 90^{\circ}$

Table V.1 Most used kernels for SVM classifier

Kernel	**function**
Linear	$K(\mathrm{x},\mathrm{x}_k)=x_k^T x$
Polynomial of degree d	$K(\mathrm{x},\mathrm{x}_k)=\left(x_k^T x+1\right)^d$
Radial basis function RBF (Gaussian)	$K(\mathrm{x},\mathrm{x}_k)=\exp\left\{-\left\|x-x_k\right\|_2^2/\sigma^2\right\}$
two layer neural	$K(\mathrm{x},\mathrm{x}_k)=\tanh\left(\kappa x_k^T+\theta\right)$; κ,θ constants

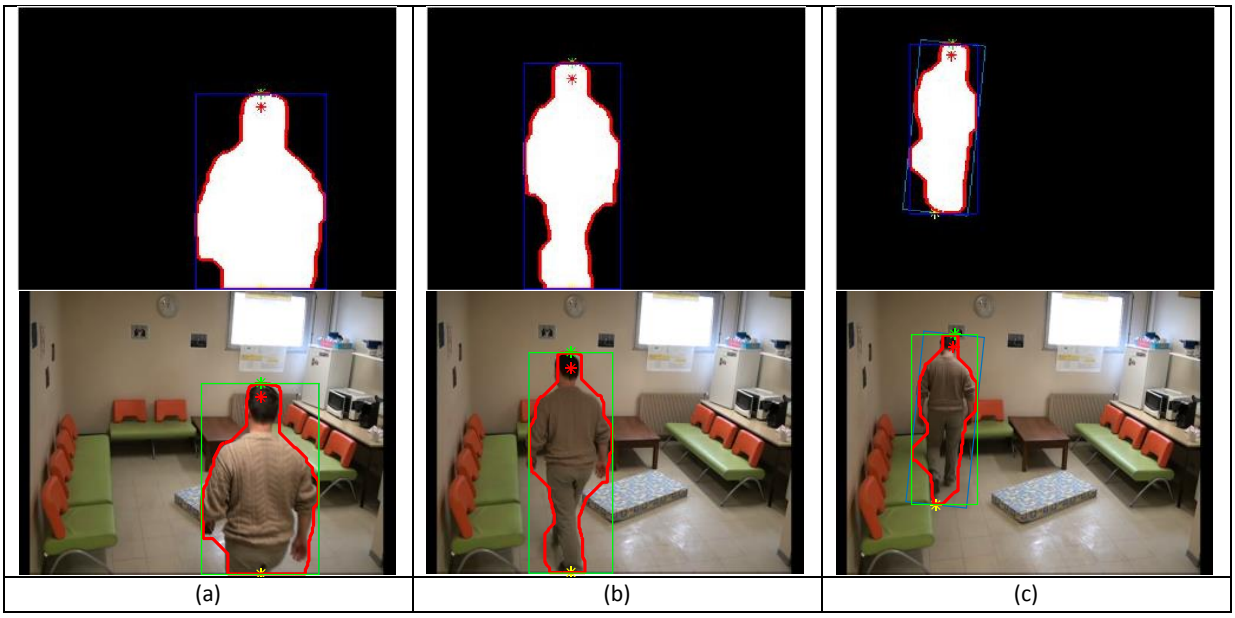
The SVM classifier takes the form represented in Eq.V.17, [49].

$$D(x)=Sign\left[\sum_{k=1}^{N}\alpha_k y_k K(x,x_k)+b\right] \qquad (V.17)$$

where α_k are positive real constants.

In our tests a RBF kernel was used with Sequential Minimal Optimization (SMO) to minimize the non-separable data problem [50].

The K-nearest neighbour is a non-parametric lazy learning algorithm. It does not make any assumptions on the underlying data distribution. The training phase of this algorithm consists of just storing all the training set $\{x_k, y_k\}_{k=1}^{N}$. In the classification phase, we search for group of κ feature vectores (neighbours) in the training set that are closest to the test object by performing a similarity measure, which is determined by computing the distance $d(x_k, z)$ between every point x_k and the test object z .



(a) (b) (c)

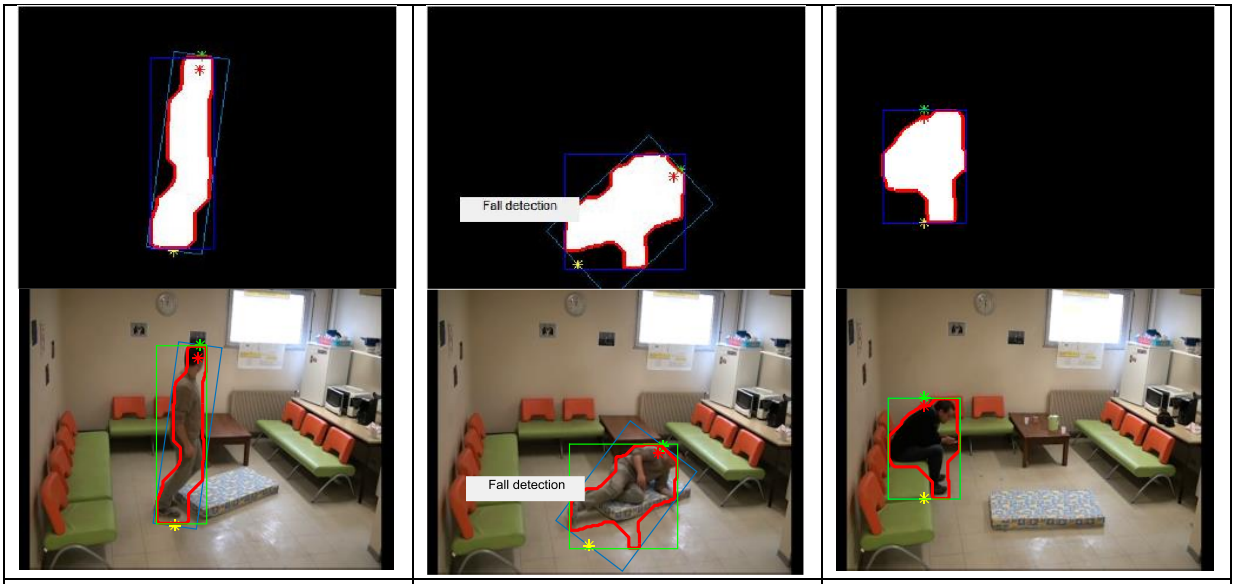


Fig.V.7. background extraction with head coordinates estimation: 21th frame from video(3); (b) 51th frame from video(3); (c) 91th frame from video(3); (d) 210th frame from video(3); (e) 230th image from video(3); (f) 11th frame from video (40)

Finally, the test object is classified based on majority voting [51]. The parameters of this algorithm were set by using an automatic hyperparameter optimization that minimized five-fold cross-validation. From that we chose $\kappa = 9$ and the standardized (weighted) Euclidean distance [52,53] as distance metric:

$$d(x, z) = \sqrt{\sum_{j=1}^{J} w_j \left(x_j - z_j\right)^2} \qquad \text{(V.18)}$$

where $w_j = 1/s_j^2$ is the inverse of the j^{th} variance.

Backpropagation neural network are multilayer feed forward networks presented by Rumelhart *et.al* [48]. The learning phase consists of minimizing the error between the input pattern and their category by adjusting the weights of the neural network. In the classification phase, for a test pattern the output is computed using the trained network, and the class can be determined by selecting the category associated with the output unit that has the largest output value. For fast learning, the Levenberg-Marquardt optimization technique [54,55] has been used along with mean squared errors (mse) as performance function. The network has two hidden layers with a size of 10 neurons. This size corresponds to the best empirically obtained result.

4.3 Accuracy of the proposed system

Accuracy is simply defined as the percentage of correct classification, given by:

$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \tag{V.19}$$

where, TP (True Positive) is the number of tests correctly classified as 'fall', TN (True Negative) is the number of tests correctly classified as 'no fall', FP (False Negative) is the number of tests incorrectly classified as 'fall' and FN (False Negative) is the number of tests incorrectly classified as 'no fall'.

Table V.2 The accuracy of our fall detection system using the three classification techniques

Classifier	RBF-SVM	KNN	BPNN
Accuracy	99.27	98.91	99.61

Because the training and the test sets was randomly chosen, the result can be slightly different in each test, but in all cases the error rate did not exceed 1,5% for all the three classifiers. This demonstrate the robustness of our fall detection system. It is worth mentioning that this error could be greatly reduced by improving the algorithm used for estimating the head position especially when the person is sitting or bending in order to compute precisely the vertical velocity of the head.

From testing different situations of fall/no fall on the L2ei dataset, we noticed that the vertical velocity of the head is the most significant and robust feature to detect a fall, this is due to its invulnerability to occlusions and to different orientations of fall. The second best feature was the height/width ratio. However this feature presents two major drawbacks, in case of the object become occluded which may generates false positives, and when the fall is totally aligned with the line of sight of the camera. Orientation of the silhouette and ferret diameter present the same drawbacks of the height/width ratio but the results were more affected and more altered. The Procrustes distance gave good results. However in some cases like arm movement or occlusion the shapes will do not match causing high Procrustes distance. Based on the dispersedness of the object when a fall happen, the change in the area does not provide a good feature for fall detection due to many situations that alter this feature, e.g. standing, sitting, getting occluded by an object or getting out from occlusion. However combining it with other features can help detecting falls and overpassing these drawbacks.

Table V.2 shows that the RBF-SVM is outperforming KNN. This is due to the fact that an RBF-SVM can be seen as improved version of KNN where we keep only the points in the frontier of classification (support vectors). Compared to BPNN, RBF-SVM has an approximate accuracy. In some tests, RBF-SVM accuracy is better than BPNN. This is due to the amount of data treated which is not sufficiently large for BPNN to converge better. Compared to the work of Charfi et al. [44], our system gave similar results with a global error less than 1%. For Rougier et al. [18], the minimum global error was about 3.8% for the multiple camera system. However this comparison is difficult due to the use of different test protocols and different datasets.

5. CONCLUSION

In this chapter, we propose an automatic fall detection system. The system detects the silhouette of the moving person then it extracts a vector of features. In addition, a FSM algorithm has been developed to estimate the position of the head. These features are then fed to three trained classifier in order to detect if a fall has been occurred or not. The results of classification shows a global error rate of less than 1.5% for KNN and less than 1% for RBF-SVM and BPNN classifiers. This demonstrates that this technique is promising. additionally the accuracy can be more enhanced by: (1) enlarging the vector of features using Hu moments, orientation of the ellipse or the mean matching cost and spatiotemporal features, (2) enlarging the training set, (3) using another parameters or kernels for these classifiers, (4) using another type of classifiers, (5) using a multiple camera system, which will constitute the basis of our future work.

REFERENCES

[1] R. Igual, C. Medrano, and I. Plaza, "Challenges, issues and trends in fall detection systems," Biomedical engineering online, vol. 12, p. 66, 2013.

[2] S. Chaudhuri, H. Thompson, and G. Demiris, "Fall detection devices and their use with older adults: a systematic review," Journal of geriatric physical therapy (2001), vol. 37, p. 178, 2014.

[3] Z. Zhang, C. Conly, and V. Athitsos, "A survey on vision-based fall detection," in Proceedings of the 8th ACM International Conference on PErvasive Technologies Related to Assistive Environments, 2015, p. 46.

[4] M. Mubashir, L. Shao, and L. Seed, "A survey on fall detection: Principles and approaches," Neurocomputing, vol. 100, pp. 144-152, 2013.

[5] M. Alwan, P. J. Rajendran, S. Kell, D. Mack, S. Dalal, M. Wolfe, et al., "A smart and passive floor-vibration based fall detector for elderly," in Information and Communication Technologies, 2006. ICTTA'06. 2nd, 2006, pp. 1003-1007.

[6] K. Chaccour, R. Darazi, A. H. el Hassans, and E. Andres, "Smart carpet using differential piezoresistive pressure sensors for elderly fall detection," in Wireless and Mobile Computing, Networking and Communications (WiMob), 2015 IEEE 11th International Conference on, 2015, pp. 225-229.

[7] H.-W. Tzeng, M.-Y. Chen, and J.-Y. Chen, "Design of fall detection system with floor pressure and infrared image," in System Science and Engineering (ICSSE), 2010 International Conference on, 2010, pp. 131-135.

[8] L. Palmerini, F. Bagalà, A. Zanetti, J. Klenk, C. Becker, and A. Cappello, "A wavelet-based approach to fall detection," Sensors, vol. 15, pp. 11575-11586, 2015.

[9] J. A. Álvarez-García, L. M. S. Morillo, M. Á. Á. de La Concepción, A. Fernández-Montes, and J. A. O. Ramírez, "Evaluating wearable activity recognition and fall detection systems," in 6th European Conference of the International Federation for Medical and Biological Engineering, 2015, pp. 653-656.

[10] O. Aziz, M. Musngi, E. J. Park, G. Mori, and S. N. Robinovitch, "A comparison of accuracy of fall detection algorithms (threshold-based vs. machine learning) using waist-mounted tri-axial accelerometer signals from a comprehensive set of falls and non-fall trials," Medical & biological engineering & computing, vol. 55, pp. 45-55, 2017.

[11] B. Y. Su, K. Ho, M. J. Rantz, and M. Skubic, "Doppler radar fall activity detection using the wavelet transform," IEEE Transactions on Biomedical Engineering, vol. 62, pp. 865-875, 2015.

[12] M. G. Amin, Y. D. Zhang, F. Ahmad, and K. D. Ho, "Radar signal processing for elderly fall detection: The future for in-home monitoring," IEEE Signal Processing Magazine, vol. 33, pp. 71-80, 2016.

[13] L. Liu, M. Popescu, M. Skubic, M. Rantz, and P. Cuddihy, "An automatic in-home fall detection system using Doppler radar signatures," Journal of Ambient Intelligence and Smart Environments, vol. 8, pp. 453-466, 2016.

[14] S. Andersson and H. Carlsson, "Brain Activity and Healthcare in the Smart Home," ed, 2014.

[15] D. De Venuto, V. Annese, M. de Tommaso, E. Vecchio, and A. S. Vincentelli, "Combining EEG and EMG Signals in a Wireless System for Preventing Fall in Neurodegenerative Diseases," in Ambient Assisted Living, ed: Springer, 2015, pp. 317-327.

[16] J. Light, T. Kalaiselvi, X. Li, and A. R. Malali, "Fall Pattern Classification from Barin Signals using Machine Learning Models," Journal of Selected Areas in Telecommunications (JSAT), Cyber Journals: Multidisciplinary Journals in Science and Technology, pp. 1-3, 2013.

[17] T. Lee and A. Mihailidis, "An intelligent emergency response system: preliminary development and testing of automated fall detection," Journal of telemedicine and telecare, vol. 11, pp. 194-198, 2005.

[18] C. Rougier, J. Meunier, A. St-Arnaud, and J. Rousseau, "Robust video surveillance for fall detection based on human shape deformation," IEEE Transactions on circuits and systems for video Technology, vol. 21, pp. 611-622, 2011.

[19] S. Wang, L. Chen, Z. Zhou, X. Sun, and J. Dong, "Human fall detection in surveillance video based on PCANet," Multimedia tools and applications, vol. 75, pp. 11603-11613, 2016

[20] L. Alhimale, H. Zedan, and A. Al-Bayatti, "The implementation of an intelligent and video-based fall detection system using a neural network," Applied Soft Computing, vol. 18, pp. 59-69, 2014.

[21] C.-L. Huang, E.-L. Chen, and P.-C. Chung, "Fall detection using modular neural networks with back-projected optical flow," Biomedical Engineering: Applications, Basis and Communications, vol. 19, pp. 415-424, 2007.

[22] S. A. Bridenbaugh and R. W. Kressig, "Laboratory review: the role of gait analysis in seniors' mobility and fall prevention," Gerontology, vol. 57, pp. 256-264, 2011.

[23] G. Baldewijns, V. Claes, G. Debard, M. Mertens, E. Devriendt, K. Milisen, et al., "Automated in-home gait transfer time analysis using video cameras," Journal of Ambient Intelligence and Smart Environments, vol. 8, pp. 273-286, 2016.

[24] A. Sixsmith and N. Johnson, "A smart sensor to detect the falls of the elderly," IEEE Pervasive computing, vol. 3, pp. 42-47, 2004.

[25] Z. Zhou, E. E. Stone, M. Skubic, J. Keller, and Z. He, "Nighttime in-home action monitoring for eldercare," in Engineering in Medicine and Biology Society, EMBC, 2011 Annual International Conference of the IEEE, 2011, pp. 5299-5302.

[26] R. Gouiaa and J. Meunier, "Human posture recognition by combining silhouette and infrared cast shadows," in Image Processing Theory, Tools and Applications (IPTA), 2015 International Conference on, 2015, pp. 49-54.

[27] G. Mastorakis and D. Makris, "Fall detection system using Kinect's infrared sensor," Journal of Real-Time Image Processing, vol. 9, pp. 635-646, 2014.

[28] E. E. Stone and M. Skubic, "Fall detection in homes of older adults using the Microsoft Kinect," IEEE journal of biomedical and health informatics, vol. 19, pp. 290-301, 2015.

[29] Z.-P. Bian, J. Hou, L.-P. Chau, and N. Magnenat-Thalmann, "Fall detection based on body part tracking using a depth camera," IEEE journal of biomedical and health informatics, vol. 19, pp. 430-439, 2015.

[30] B. Bonnechere, B. Jansen, P. Salvia, H. Bouzahouene, L. Omelina, J. Cornelis, et al., "What are the current limits of the Kinect sensor," in Ninth International Conference on Disability, Virtual Reality and Associated Technologies (ICDVRAT), 2012, pp. 287-294.

[31] C. Rougier, J. Meunier, A. St-Arnaud, and J. Rousseau, "3D head tracking for fall detection using a single calibrated camera," Image and Vision Computing, vol. 31, pp. 246-254, 2013.

[32] L. Hazelhoff and J. Han, "Video-based fall detection in the home using principal component analysis," in International Conference on Advanced Concepts for Intelligent Vision Systems, 2008, pp. 298-309.

[33] K. Sehairi, F. Chouireb, and J. Meunier, "Comparative study of motion detection methods for video surveillance systems," Journal of Electronic Imaging, vol. 26, pp. 023025-023025, 2017.

[34] G. D. Sergio and V. P. Javier, "Simplified SOM-neural model for video segmentation of moving objects," in Neural Networks, 2009. IJCNN 2009. International Joint Conference on, 2009, pp. 474-480.

[35] F. Naini, M. Cobourne, F. McDonald, and A. Donaldson, "The influence of craniofacial to standing height proportion on perceived attractiveness," International journal of oral and maxillofacial surgery, vol. 37, pp. 877-885, 2008.

[36] I. L. Dryden and K. V. Mardia, Statistical Shape Analysis: With Applications in R: John Wiley & Sons, 2016.

[37] J. C. Gower, "Generalized procrustes analysis," Psychometrika, vol. 40, pp. 33-51, 1975.

[38] S. M. Shin, Y.-M. Kim, N.-R. Kim, Y.-S. Choi, S.-B. Park, and Y.-I. Kim, "Statistical shape analysis-based determination of optimal midsagittal reference plane for evaluation of facial asymmetry," American Journal of Orthodontics and Dentofacial Orthopedics, vol. 150, pp. 252-260, 2016.

[39] W. Cho, S. Kim, S. Park, and M. Lee, "Human action classification using procrustes shape theory," in SPIE/IS&T Electronic Imaging, 2015, pp. 94050T-94050T-7.

[40] W. Cho, S. Kang, S. Kim, and S. Park, "Human Action Recognition Using Product Manifold Theory and Procrustes Shape Analysis," in Computer Science and its Applications, ed: Springer, 2015, pp. 831-836.

[41] S. Jahanbin, R. Jahanbin, and A. C. Bovik, "Passive three dimensional face recognition using iso-geodesic contours and procrustes analysis," International journal of computer vision, vol. 105, pp. 87-108, 2013.

[42] L. Wang, H. Ning, T. Tan, and W. Hu, "Fusion of static and dynamic body biometrics for gait recognition," IEEE Transactions on circuits and systems for video technology, vol. 14, pp. 149-158, 2004.

[43] N. Noury, P. Rumeau, A. Bourke, G. ÓLaighin, and J. Lundy, "A proposal for the classification and evaluation of fall detectors," Irbm, vol. 29, pp. 340-349, 2008.

[44] I. Charfi, J. Miteran, J. Dubois, M. Atri, and R. Tourki, "Definition and performance evaluation of a robust svm based fall detection solution," in Signal Image Technology and Internet Based Systems (SITIS), 2012 Eighth International Conference on, 2012, pp. 218-224.

[45] G. Baldewijns, G. Debard, G. Mertes, B. Vanrumste, and T. Croonenborghs, "Bridging the gap between real-life data and simulated data by providing a highly realistic fall dataset for evaluating camera-based fall detection algorithms," Healthcare technology letters, vol. 3, pp. 6-11, 2016.

[46] V. Vapnik, The nature of statistical learning theory: Springer science & business media, 2013.

[47] P.-N. Tan, Introduction to data mining: Pearson Education India, 2006.

[48] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," Cognitive modeling, vol. 5, p. 1, 1988.

[49] J. A. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," Neural processing letters, vol. 9, pp. 293-300, 1999.

[50] R.-E. Fan, P.-H. Chen, and C.-J. Lin, "Working set selection using second order information for training support vector machines," Journal of machine learning research, vol. 6, pp. 1889-1918, 2005.

[51] X. Wu, V. Kumar, J. R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, et al., "Top 10 algorithms in data mining," Knowledge and information systems, vol. 14, pp. 1-37, 2008.

[52] M. Greenacre, Correspondence Analysis in Practice: CRC Press, 2007.

[53] V. Kumar, J. K. Chhabra, and D. Kumar, "Impact of distance measures on the performance of clustering algorithms," in Intelligent Computing, Networking, and Informatics, ed: Springer, 2014, pp. 183-190.

[54] D. W. Marquardt, "An algorithm for least-squares estimation of nonlinear parameters," Journal of the society for Industrial and Applied Mathematics, vol. 11, pp. 431-441, 1963.

[55] H. Yu and B. M. Wilamowski, "Levenberg–marquardt training," Industrial Electronics Handbook, vol. 5, p. 1, 2011.

Conclusion

“Success is a science; if you have the conditions, you get the result.”

Oscar Wilde

Conclusion and perspectives

The work presented in this thesis was conducted in telecommunications, signals and systems laboratory at the University of Laghouat, in cordination with the image processing laboratory of the computer science and operations research department at the University of Montreal.

The aim was to implement a complete real time video surveillance system that detect moving objects, classify them and recognize their activities. Concurrently, evaluate existing methods, to choose appropriately the best method and develop new algorithms and techniques to overcome the problems of the existing ones.

For that, we started our work by reviewing the state of the art of video surveillance techniques: motion detection and segmentation, classification and identification and at last action recognition and behaviour understanding.

From this study, we have analysed that many motion detection techniques have not been evaluated on big and challenging datasets. For that, a comparative study was performed to define the best method under different situations such as simple indoor/outdoor scenes (baseline), pan, tilt, zoom camera, thermal camera (far-infrared camera), bad weather conditions, dynamic backgrounds, intermittent object motion, camera jitter, presence of shadows, night videos, low framerate and case of turbulence (near infrared camera). A detailed description has been done to help the user to choose the apprpriate method for his application.

To fill the lack of studies in segmentation techniques for motion detection methods and to fully automatize our video surveillance, a second comparative study was conducted. The result of this method can help the user to define the appropriate threshold method for the motion detection techniques.

In the second part of our work, we focused on developing video surveillance systems. Two systems have been designed. The first system is a real time system for detecting and recognizing actions. It has been achieved by implementing a finite state machine FSM. A parallel design has been developed to show the results in RGB. The system has been implemented on Virtex II FPGA, using high level language (Handel-

C). It can run at 53,83 Mhz (i.e. >128 fps for image with resolution of 720×576). The resource consumption of the hardware circuit has not exceeded 48%.

The second system consists of intelligent video-based fall detection for elder people. This system contains many stages, including a background modelling stage, a foreground extraction stage, a segmentation stage, a pre-processing stage, a feature extraction stage and at last a classification stage. A new technique was developed to estimate the head coordinates in order to compute its vertical velocity. The system uses machine learning techniques to detect and identify falls. For that more than 2700 frames were labelled and three different classifiers were trained. The system is able to recognize falls with an accuracy of 99.61%.

As future work, we propose:

- The use of spatio-temporal features for action recognition.
- The development of new 3D features for action recognition.
- The use of 3D and LSTM deep convolution neural networks for automated feature extraction and recognition.
- The implementation of these techniques on recent FPGA SoCs and GPUs.
- The development of large dataset for fall detection to be used for learning.

Appendix

“I seem to have been only like a boy playing on the seashore, and diverting myself in now and then finding a smoother pebble or a prettier shell than ordinary, whilst the great ocean of truth lay all undiscovered before me.”

Isaac Newton

Change Detection.net (CDnet) dataset

CDnet video datasets (CDnet2012 & Cdnet2014) consist of diverse set of videos regrouped in eleven categories, each category regroups four to six videos with different challenging problems, and each video frame is accompanied by accurate ground-truth segmentation and annotation of change/motion areas, the description of each category is given below:

1. **Baseline:** This category contains four videos, two indoor and two outdoor. These videos represent a mixture of mild challenges typical of the next 4 categories. Some videos have subtle background motion, others have isolated shadows, some have an abandoned object and others have pedestrians that stop for a short while and then move away. These videos are fairly easy, but not trivial, to process, and are provided mainly as reference.

2. **Dynamic Background:** There are six videos in this category depicting outdoor scenes with strong (parasitic) background motion. Two videos represent boats on shimmering water, two videos show cars passing next to a fountain, and the last two depict pedestrians, cars and trucks passing in front of a tree shaken by the wind.

3. **Camera Jitter:** This category contains one indoor and three outdoor videos captured by unstable (e.g., vibrating) cameras. The jitter magnitude varies from one video to another.

4. **Shadows:** This category consists of two indoor and four outdoor videos exhibiting strong as well as faint shadows. Some shadows are fairly narrow while others occupy most of the scene. Also, some shadows are cast by moving objects while others are cast by trees and buildings.

5. **Intermittent Object Motion:** This category contains six videos with scenarios known for causing "ghosting" artifacts in the detected motion, i.e., objects move, then stop for a short while, after which they start moving again. Some videos include still objects that suddenly start moving, e.g., a parked vehicle driving away, and also abandoned objects. This category is intended for testing how various algorithms adapt to background changes.

6. **Thermal:** In this category, five videos (three outdoor and two indoor) have been captured by far-infrared cameras. These videos contain typical thermal artifacts such as

heat stamps (e.g., bright spots left on a seat after a person gets up and leaves), heat reflection on floors and windows, and camouflage effects, when a moving object has the same temperature as the surrounding regions.

7. **Challenging Weather:** This category contains 4 outdoor videos showing low-visibility winter storm conditions. This includes two traffic scenes in a blizzard, cars and pedestrians at the corner of a street and people skating in the snow. These videos present a double challenge: in addition to snow accumulation, the dark tire tracks left in the snow have potential to cause false positives.

8. **Low Frame-Rate:** In this category 4 videos, all recorded with IP cameras, are included. The frame rate varies from 0.17 fps to 1 fps due to limited transmission bandwidth. By nature, these videos show "erratic motion patterns" of moving objects that are hard (if not impossible) to correlate. Optical flow might be ineffective for these videos. One sequence is particularly challenging (port 0 17fps), which shows boats and people coming in and out of a harbor, as the low framerate accentuates the wavy motion of moored boats causing false detections.

9. **Night:** This category has 6 motor traffic videos. The main challenge is to cope with low-visibility of vehicles yet their very strong headlights that cause over saturation. Headlights cause halos and reflections on the street.

10. **PTZ:** We included 4 videos in this category: one video with a slow continuous camera pan, one video with an intermittent pan, one video with a 2-position patrol-mode PTZ, and one video with zoom-in/zoom-out. The PTZ category by itself requires different type of change detection techniques in comparison to static camera videos.

11. **Air Turbulence:** This category contains 4 videos showing moving objects depicted by a near-infrared camera at noon during a hot summer day. Since the scene is filmed at a distance (5 to 15 km) with a telephoto lens, the heat causes constant air turbulence and distortion in frames. This results in false positives. The size of the moving objects also varies significantly from one video to another. The air turbulence category presents very similar challenges to those arising in long-distance remote surveillance applications.