Technical University of Crete

Detection and Semantic Analysis of Objects and Events through Visual Cues

Thesis submitted in partial fulfillment of
the requirements for the degree of
Doctor of Philosophy

by

Konstantinos Makantasis

Chania, Greece, June 2016

The thesis is approved by the following jury:

Nikolaos Matsatsinis (supervisor) .....
Professor, School of Production Engineering and Management
Technical University of Crete

Anastasios Doulamis (member of committee in charge) .....
Lecturer, School of Rural and Surveying Engineering
National Technical University of Athens

Michalis Zervakis (member of committee in charge) .....
Professor, School of Electronic and Computer Engineering
Technical University of Crete

Nicolas Tsapatsoulis .....
Associate Professor, Department of Communication and Internet Studies
Cyprus University of Technology

Panagiotis Partsinevelos .....
Assistant Professor, School of Mineral Resources Engineering
Technical University of Crete

Stelios Tsafarakis .....
Assistant Professor, School of Production Engineering and Management
Technical University of Crete

Nikolaos Grammalidis .....
Senior Researcher (Grade B), Informatics and Telematics Institute
Center of Research and Technology Hellas

To my family

"In every conceivable manner, the family is link to our past, bridge to our future."

— Alex Hailey

To the memory of

David Marr (1945 – 1980),
who set the foundations of computer vision,

and

Aaron Swartz (1986 – 2013),
who fought and finally died for open access to science.

No computer has ever been designed that
is ever aware of what it's doing;
but most of the time, we aren't either.

— Marvin Minsky

ACKNOWLEDGMENTS

This thesis is a monograph. However, I believe that a PhD thesis is not the effort of an individual, but the final outcome of a synergy. For this reason, I would like to thank, in these initial pages, the people who involved in a way or another with my PhD thesis, and provided psychological, scientific and technical support.

First, and foremost, I would like to thank my supervisor Prof. Nikolaos Matsatsinis for his interest in my work, for his supervision and availability, and for leaving me a considerable amount of freedom in my research. His role was two-fold; that of an academic teacher and a personal research advisor. Prof. Matsatsinis guided me through my doctoral studies, infused me a wide variety of knowledge and helped me to mature as a researcher.

I am deeply in debt with Tasos Doulamis, who was the initiator of this thesis and my supervisor during the first year of my PhD. He was not only an advisor, but also my closest research colleague. I had the privilege to appreciate his natural disposition to put everyone at ease, and recognize his high scientific quality. He provided me with visionary questions – if I could have answered more of them, maybe I was a famous machine learning and computer vision practitioner today – and simple yet illuminating answers to my questions.

Furthermore, I have appreciated the trust of Prof. Mihalis Zervakis to be a member of committee in charge and I really thank him for believing in me, from the very start, during my preliminary steps in research.

I would like to thank Nikolas Tsapatsoulis for showing me how to organize the work in the framework of a complicated project and Panagiotis Partsinevelos who was the first man that trusted me and my work and introduced me to academic research. I also thank Nikos Grammalidis, I met him at a conference and he may determine my future research work using higher order tensors, and Stelios Tsafarakis who put me up against challenging operational research problems.

I also appreciate the excellent contribution and help of Nikos Doulamis. His experience, patience and guidance was of utmost importance for the completion of this thesis. Finally, I feel obliged to express my gratitude to Konstantinos Karantzalos for introducing me to the world of computer vision beyond the visual spectrum.

A very special acknowledgment goes to Maria Kontorinaki for all her love and support and for the infinite inspiration that she has given me to deal with different things ranging from music to philosophy to mathematics. I would also like to acknowledge her family who treated me like family and never made me feel like a stranger.

I also thank a number of researchers at the Technical University of Crete. Especially, I would like to thank Alexandros Georgogiannis and Antonis Nikitakis – during our occasional coffees I could find interested and competent minds in my disposal– and Eftychios Protopapadakis who is my

co-author in many publications, and after all he was my *roommate* at the university office for more than four years.

Last but not least, I would like to thank my family and all my friends for their psychological support – I really thank you all.

Chania, Greece, June 2016

SHORT BIOGRAPHY

Konstantinos Makantasis received his computer engineering diploma from the Technical university of Crete (TUC, Greece) and his Master degree from the same school (DPEM, TUC). His diploma thesis entitled "Human face detection and tracking using AIBO robots", while his master thesis entitled "Persons' fall detection through visual cue". He is mostly involved and interested in computer vision, both for visual spectrum (RGB) and hyperspectral data, and in machine learning / pattern recognition and probabilistic programming. He has more than 20 publications in international journals and conferences on computer vision, signal and image processing and machine learning. He has been involved for more than 6 years as a researcher in numerous European and national competing research programs (Interreg, FP7, Marie Curie actions) towards the design, development and validation of state-of-the-art methodologies and cutting-edge technologies in data analytics and computer vision.

ABSTRACT

This dissertation counts in total ten technical chapters, plus a conclusion chapter. Technical chapters are organized in three parts; each one of them is dedicated to a different aspect of semantic visual content analysis.

The first part consists of four chapters and titled "*From Objects to Events*". As the title suggests, in this part, we investigate how the information about objects in a scene can be available as a basis for event understanding. There are three different technical chapters, in which we try to address three different real-world problems; the development of (i) a supportive vision based system for detecting in real-time elderly and/or patients fall in indoor environments, (ii) a maritime security vision based system and (iii) a surveillance system for activity recognition in industrial workflow.

The second part titled "*From Unstructured Visual Content to Objects*". In this part we investigate how the visual content that is stored in distributed and heterogeneous Internet databases can be, initially, organized, and then utilized towards objects documentation. Specifically, in this part we propose (i) a method for retrieving and dynamically clustering user generated photographs available over the web and (ii) and online image indexing scheme.

The third and last part titled "*Beyond the Visual Spectrum*" and focuses on visual content analysis using thermal and hyperspectral data. There are two technical chapters; the first one presents an algorithm for background subtraction applied on thermal video streams, while the second one presents a method for material recognition using hyperspectral images.

We approach each one of the aforementioned problems through the *levels of understanding* framework. Initially, we formulate in detail the problem at hand along with its constraints and specifications, explaining what computations will do and why they will do it. Then, we proceed with proposed solution design and implementation, where we describe in detail the tools for developing the proposed solutions, the input and output of the system as well as, all intermediate representations of visual information. Finally, we evaluate proposed solutions performance on both synthetic and real-world data.

Journal Papers

K.Makantasis, E.Protopapadakis, A.Doulamis, N.Doulamis, N.Matsatsinis "3D measures exploitation for a monocular semi-supervised fall detection system" Multimedia Tools and Applications (April 2015), 1-33

K.Makantasis, E.Protopapadakis, A.Doulamis, N.Matsatsinis "Semi-supervised vision-based maritime surveillance system using fused visual attention maps" Multimedia Tools and Applications (March 2015), 1-28

K.Makantasis, A.Doulamis, N.Doulamis, M.Ioannides " In the wild image retrieval and clustering for 3D cultural heritage landmarks reconstruction" Multimedia Tools and Applications (August 2014), 1-37

G.Kyriakaki, A.Doulamis, N.Doulamis, M.Ioannides, K.Makantasis, E.Protopapadakis, A.Hadjiprocopis et al. "4D Reconstruction of Tangible Cultural Heritage Objects from Web-Retrieved Images." International Journal of Heritage in the Digital Era 3, no. 2 (2014): 431-452

P.Partsinevelos, K.Papadakis, K.Makantasis. "Spatiotemporal graph queries on geographic databases under a conceptual abstraction scale." Geo-spatial Information Science 17.2 (2014): 110-118.

P.Partsinevelos, E.Stamboliadis, K.Makantasis. "Image based mineral liberation simulation incorporating experimental grinding models." Canadian Metallurgical Quarterly 51.4 (2012): 383-389.

K.Makantasis, A. Doulamis. "3D Measures Computed in Monocular Camera System and SVM-based Classifier for Humans Fall Detection." TMC Academic Journal, Vol. 7, Issue 2, pp 1-14, Feb/Mar 2012

Conference Papers

K.Makantasis, A.Doulamis, N.Doulamis, K.Psychas, "Deep learning based human recognition in industrial workflows" to be appeared in International Conference on Image Processing (ICIP 2016), IEEE

A.Nikitakis, I.Papaefstathiou, K.Makantasis, A.Doulamis, (2016, March). "A novel background subtraction scheme for in-camera acceleration in thermal imagery". In 2016 Design, Automation and Test in Europe Conference and Exhibition (DATE) (pp. 1497-1500). IEEE.

N.Doulamis, A.Doulamis, K.Makantasis, K.Karantzalos, K.Loupos,(2015, July). "Micro-scale thermal behavioral analysis for active evacuation routes". In Proceedings of the 8th ACM International Conference on PErvasive Technologies Related to Assistive Environments (p. 50). ACM.

K.Makantasis, K.Karantzalos, A.Doulamis, K.Loupos. "Deep Learning-Based Man-Made Object Detection from Hyperspectral Data." Advances in Visual Computing. Springer International Publishing, 2015. 717-727.

K.Makantasis, A.Doulamis, K.Loupos. "Variational Inference for Background Subtraction in Infrared Imagery." Advances in Visual Computing. Springer International Publishing, 2015. 693-705.

K.Makantasis, E.Protopapadakis, A.Doulamis, N.Doulamis, K.Loupos, (2015, September). "Deep convolutional neural networks for efficient vision based tunnel inspection". In Intelligent Computer Communication and Processing (ICCP), 2015 IEEE International Conference on (pp. 335-342). IEEE.

K.Makantasis, K.Karantzalos, A.Doulamis, N.Doulamis, (2015, July). "Deep supervised learning for hyperspectral data classification through convolutional neural networks". In Geoscience and Remote Sensing Symposium (IGARSS), 2015 IEEE International (pp. 4959-4962). IEEE.

K.Makantasis, A.Doulamis, N.Doulamis, M.Ioannides, N.Matsatsinis "Content-Based Filtering for Fast 3D Reconstruction from Unstructured Web-Based Image Data", EUROMED 2014

K.Makantasis, A.Doulamis, N.Doulamis. "Vision-based maritime surveillance system using fused visual attention maps and online adaptable tracker." Image Analysis for Multimedia Interactive Services (WIAMIS), 2013 14th International Workshop on. IEEE, 2013

K.Makantasis, A.Doulamis, N.Doulamis. "A non-parametric unsupervised approach for content based image retrieval and clustering." Proceedings of the 4th ACM/IEEE international workshop on Analysis and retrieval of tracked events and motion in imagery stream. ACM, 2013

M.Ioannides, A.Hadjiprocopis, N.Doulamis, A.Doulamis, E.Protopapadakis, K.Makantasis, P.Santos et al. "Online 4d Reconstruction Using Multi-Images Available Under Open Access." ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences 1, no. 1 (2013): 169-174

K.Makantasis, A.Doulamis, N.Matsatsinis. "Student-t background modeling for persons' fall detection through visual cues." Image Analysis for Multimedia Interactive Services (WIAMIS), 2012 13th International Workshop on. IEEE, 2012

K.Makantasis, E.Protopapadakis, A.Doulamis, L. Grammatikopoulos, C.Stentoumis,(2012, January). Monocular camera fall detection system exploiting 3d measures: a semi-supervised learning approach. In Computer Vision–ECCV 2012. Workshops and Demonstrations (pp. 81-90). Springer Berlin Heidelberg

K.Makantasis, A.Doulamis, N.Matsatsinis. "3D Measures Computed in Monocular Camera System for Fall Detection." INFOCOMP 2012, The Second International Conference on Advanced Communications and Computation. 2012

E.Protopapadakis, A.Doulamis, K.Makantasis, A. Voulodimos (2012, October). A Semi-supervised Approach for Industrial Workflow Recognition. In INFOCOMP 2012, The Second International Conference on Advanced Communications and Computation (pp. 155-160)

CONTENTS

LIST OF TABLES

1 INTRODUCTION

Humans receive the great majority of information about their environment through sight. But, *what* is sight or otherwise *what* does it mean to see? Physicists merely answer to this question focusing on the surrounding environment of a seeing organism. They can trace the route of light radiating outwards from light sources as it is partially absorbed and partially reflected off of bodies in the environment until it arrives at the light sensitive organs of an organism. Physiologists are focusing on another part of seeing. They investigate light's passage through the pupils and the lens through which it impacts the rods and cones of the retina, setting off the electronic transmission of the impact via a set of electrochemical switches by means of which it ends up in the visual cortex at the back of the brain. Beyond these technical details, the plain answer (and Aristotle's too) is to know what is where by looking. Unfortunately, looking is not some kind of direct perception of reality. Actually, our brains are constantly interpreting, correcting and giving structures to the visual input from our eyes. In other words, vision is the information-processing task of discovering from images what is present in the world, and where it is.

The capability of knowing what is where in the world suggests that our brains must somehow be capable of representing this information. The study of vision must therefore include not only the investigation of how to extract from images the various aspects of the world that are useful to us, but also an inquiry into the nature of the internal representations by which we capture this information and thus make it available as a basis for decisions about our thoughts and actions. This duality – the representation and the processing of information – lies at the heart of most information-processing tasks. Perceptual psychologists have spent decades trying to understand and explore vision process. However, a complete solution to this puzzle remains elusive (Palmer, 1999; Livingstone, 2002).

Even though there is no a clear picture of how vision works, exploitation of visual content is a key component for building artificial intelligence systems, and computer vision is considered as one of the most active research fields in information technology. Since its infancy, computer vision aroused great enthusiasm and expectations among the computer scientists and engineers. According to one well-known story, in 1966, Marvin Minsky at MIT asked his undergraduate student Gerald Jay Sussman to "spend the summer linking the camera to a computer and getting the computer to describe what is saw" (Crevier, 1993). During the early days of artificial intelligence, it was believed that the *cognitive* – internal beliefs and knowledge leading to logic reasoning – parts of intelligence were intrinsically more difficult than the *perceptual* – understanding of directional concepts to organize the surrounding space and discrimination, sorting, organization, storing and recalling already presented information – components (Boden, 2006). Since then, computer vision researchers have developed well-defined mathematical techniques and robust real-world applications for several computer vision problems, such as object detection and recognition, motion tracking, semantic analysis of visual content.

However, despite these advances, the dream of having a computer interpret an image at the same level as a two-year old remains elusive. Thus, the crucial question remains; *Why is vision so difficult*? In part, it is because vision is an inverse problem, in which we seek to recover some unknown given insufficient information to fully specify the solution; i.e. to recover the three-dimensional structure of the world from images and to use this as a stepping stone towards full scene understanding (Szeliski, 2010).

Considering the aforementioned limitations and restrictions, in this thesis we adopt a typical engineering approach to the study of vision problems. We think back from the problem at hand to suitable techniques. In other words, we propose and develop problem oriented solutions and emphasize the importance of coupling experimental and theoretical work; without close interaction with experiments, theory is very likely to be sterile. We follow David Marr's philosophy to frame and solve vision problems (Marr et al., 2010). Firstly, we come up with a detailed problem definition and decide on its constraints and specifications – *problem formulation* –, then, we try to figure out possible approaches to the problem at hand and find out techniques that are known to work – *literature review*. Secondly, we design and describe in detail our approach – *our contribution/algorithm*. Finally, we select the appropriate tools for developing our solution – *algorithm manifestation* – and evaluate its performance on realistic data, both synthetic, which are used to verify correctness and analyze noise sensitivity, and real-world data typical of the way the solution will, finally, be used.

The aforementioned approach is related to David Marr's *levels of understanding* framework (see Section 1.1), which requires the combination of (i) a careful analysis of the problem specification and known constraints from image formation and priors – the scientific and statistical approaches – with (ii) efficient and robust algorithms – the engineering approach – to design successful vision solutions. This framework laid the foundations for approaching computer vision problems and it remains as useful as it was 35 years ago, when it was firstly issued.

1.1 LEVELS OF UNDERSTANDING

David Marr's *levels of understanding* (Marr et al., 2010) is a framework for studying and understanding visual perception. In this framework, vision is considered as a process, which proceeds by constructing a set of representations, starting from a description of the input image, and culminating with a description of three-dimensional objects in the surrounding environment. The twin strands of *process* and *representation* are both key aspects in this framework and consist the first level of understanding – *computation* level.

As a *representation* considered a formal system for making explicit certain entities or types of information, together with a detailed specification of how the system does this. The result of using a representation is a *description* of a given entity. Every representation implies a trade-off; makes certain information explicit at the expense of information that is pushed into the background and may be quite hard to recover. This is a crucial issue, due to the fact that entities' representation can greatly affect how easy it is to do different things; e.g. it is easy to add, to subtract and even to multiply if the Arabic numeral representation is used, but it is not at all easy to do these things with Roman numerals. This is a key reason why the Roman culture failed to develop mathematics in the way the earlier Arabic cultures

Computation Level	Algorithm Level	Hardware Level
What is the goal of the computation and why it is appropriate, and what is the logic of the strategy by which it can be carried out?	How can these computations be implemented, what is the representation of the input and output, and what is the algorithm of the transformation?	How can the representation and the algorithm be realized physically?

Table 1.1: Levels of understanding framework.

had. The usefulness of a representation depends upon how well suited it is to the purpose for which it is used. We can only do what is possible and proceed from there toward what is desirable. Thus we arrive at the idea of a sequence of representations, starting with descriptions that could be obtained straight from an image but that are carefully designed to facilitate the subsequent recovery of gradually more high-level and objective, physical properties about an object's properties.

Although, there are several levels at which one can understand a *process*, the most abstract one is the level of *what* the process does and *why*. Thus part of this first level is something that might be characterized as what is being computed, while the other half has to do with the question of why this computation is performed. The important features of this level of understanding are (i) that it contains separate arguments about what is computed and why and (ii) that the resulting operation is defined uniquely by the constraints it has to satisfy.

The second level of understanding relies on the description of an algorithm by which the computations may actually be accomplished – *algorithm* level. While the computations level specifies what and why, this level specifies *how*. There are several points that must be addressed here. Firstly, there is usually a wide choice of candidate representations. Secondly, the choice of a suitable algorithm often depends rather critically on the particular representation that is employed. And, finally, even for a given fixed representation, there are often several possible algorithms for carrying out the same process. On the one hand, which one is chosen will usually depend on any particular desirable or undesirable characteristics that the algorithms may have, and on the other, this choice is inherently depended on the type of machinery in which the algorithm is to be embodied physically.

The third level is that of the device in which the process is to be realized physically – *hardware* level. When reach to this level it is crucial that, once again, the same algorithm may be implemented in quite different technologies. Some styles of algorithm will suit some physical substrates better than others; e.g. serial and parallel implementations. However, the hardware level is outside of the scope of this thesis; we focus exclusively on the computation and algorithm levels.

To summarize, there are three different levels at which an information-processing device must be understood before one can be said to have understood it completely – see Table 1.1. At the first level, the level of computation, is the abstract computational theory, in which the performance of the device is characterized as a mapping from one kind of information to another. The abstract properties of this mapping are defined precisely, and its appropriateness and adequacy for the task at hand are demonstrated. At the second level, the algorithm level, is the choice of representation for the input and output and the algorithm to be used to transform one into the other. And

at the third level, hardware level, are the details of how the algorithm and representation are realized physically.

1.2 THESIS ORGANIZATION

This dissertation counts in total ten technical chapters, plus a conclusion chapter. Technical chapters are organized in three parts; each one of them is dedicated to a different aspect of semantic visual content analysis.

The first part titled "*From Objects to Events*". As the title suggests, in this part, we investigate how the information about objects in a scene can be available as a basis for event understanding. There are three technical chapters, in which we try to address three different real-world problems. The first chapter is dedicated to the development of a supportive vision based system for detecting in real-time elderly and/or patients fall in indoor environments. In the second chapter we propose a maritime security vision based system, while in the third chapter we present a surveillance system for activity recognition in industrial workflow.

The second part contains two technical chapters and titled "*From Unstructured Visual Content to Objects*". In this part we investigate how the visual content that is stored in distributed and heterogeneous Internet databases can be, initially, organized, and then utilized towards objects documentation. Specifically, these chapters propose a method for retrieving and dynamically indexing user generated photographs available over the web. We focus on photographs that depict cultural heritage assets and we show how our retrieval and indexing system can facilitate e-documentation tools, such as 3D reconstruction.

The third and last part titled "*Beyond the Visual Spectrum*" and focuses on visual content analysis using thermal and hyperspectral data. It consists of two chapters; the first one presents a background subtraction algorithm, specifically designed to be applied on thermal video streams, while the second one presents a method for material recognition using hyperspectral images.

We approach each one of the aforementioned problems through the *levels of understanding* framework – as mentioned before, the hardware level is outside of the scope of this thesis, thus, we focus exclusively on the first two levels. Initially, we formulate in detail the problem at hand along with its constraints and specifications, explaining what computations will do and why they will do it. Then, we proceed with proposed solution design and implementation, where we describe in detail the tools for developing the proposed solutions, the input and output of the system as well as, all intermediate representations of visual information. Finally, we evaluate proposed solutions performance on both synthetic and real-world data.

Part I

FROM OBJECTS TO EVENTS

In this part, our main intention is to focus attention on important aspects of objects and events and their relationship with computer vision, and, then, to investigate how information about objects in a scene can be available as a basis for event understanding. The outcome of this investigation, enable us to analyze three different real-world problems and propose computer vision solutions for each one of them. We start with a supportive vision based system for detecting humans' falls in indoor environments. We continue with a maritime security vision based system, and finally, we present a surveillance system for activity recognition in industrial workflow.

2

OBJECTS AND EVENTS

2.1 THE NOTION OF OBJECT

The notion of "object" is closely related to human and computer vision. Cavanagh in (Cavanagh, 1992) states:

> *"the goal of vision is to inform us of the identity of objects in view and their spatial positions"*,

while Ballard and Brown (Ballard and Brown, 1982), in one of the most influential works on computer vision, mention:

> *"computer vision is the construction of explicit, meaningful descriptions of physical objects from images"*.

Although, there is no hint here that vision may consist of more than object *identification*, our everyday experience suggests that, behind identifying objects, vision is also used to track motion, detect changes in a scene, estimate depth of surfaces and texture. No matter what the visual task is, the notion of "object" is always present.

However, the notion of "object" is so deeply ingrained in our language that its logical status is rarely questioned. If the question does arise then it is usually addressed by invoking "features". For this reason, the research interest of computer vision community has been focus, to a great extend, on encoding the visual content by describing *local* and *global* properties of objects in terms of features.

A local property of an object can be represented by a feature, which is a single piece of information located on a specific point or a small region, describing a rather simple, but ideally distinctive property of the object's projection to the image plane. Examples for local features of an object are the average color or intensity value of a pixel or small region. Features describing the local properties of objects should be invariant to illumination changes, scale, rotation and noise, but, in general, this cannot be reached due to the simpleness of the features itself. Therefore, several features of a single point or a small region are combined and a more complex description of objects' local properties referred to as *descriptors* are obtained.

On the other hand, global properties of objects can be represented by features that try to cover the visual information of the whole image. This varies from statistical estimates, such as background subtraction techniques (Bouwmans et al., 2010), to image projection approaches, i.e., subspace methods such as Principal Component Analysis (Jolliffe, 2002), Independent Component Analysis (Hyvärinen et al., 2004) or Non-negative Matrix Factorization (Lee and Seung, 1999). The main idea of such methods is to project the original data onto a subspace that represents the data optimally according to a predefined criterion, which is usually depended on the specific application at hand.

Feature-based representations of objects, regardless of whether they were generated by exploiting local and/or global objects' properties, are used to learn classifiers capable of discriminating and detecting the presence of

objects of interest in a scene. Classifiers can range from simple rule based approaches to more sophisticated pattern recognition methods such as neural networks (Rojas, 2013) and kernel machines (Cortes and Vapnik, 1995a).

2.1.1 *Local Properties of Objects*

The local properties of objects correspond to information located on a specific point or a small region. Methods that find points or regions of interest are called *detectors*. The currently most popular detectors can be roughly divided into two broad categories; i) *corner-based* and ii) *region-based* detectors. Corner-based detectors locate points of interest and regions which contain a lot of image structure, however they are not suited for uniform regions and regions with smooth transitions. On the contrary, region-based detectors regard local blobs of uniform brightness as the most salient aspects of an image and are therefore more suited for the latter.

The most popular detector is the corner-based one of Harris and Stephens (Harris and Stephens, 1988). It exploits the first derivatives of image intensity to deliver a large number of interest points with sufficient repeatability (Schmid et al., 2000). The main advantage of this detector is the low computational cost. However, it determines only the spatial locations of the interest points without any interesting region properties such as scale or orientation. Hessian matrix-based detectors (second derivatives of intensity) are region-based giving strong responses on blobs and ridges. However, they show only rotational invariance properties (Bay et al., 2008). Lindeberg in (Lindeberg, 1998) intensely studied the scale-space properties of blobs. Based on this work the local extrema of the scale normalized Laplacian (Burt and Adelson, 1983) can be used as a scale selection criterion allowing the aforementioned Harris and Hessian detectors to have scale invariance properties. Consequently, in the literature they are often referred as Harris-Laplace or Hessian-Laplace detectors. Mikolajczyk and Schmid (Mikolajczyk and Schmid, 2002) proposed an extension of the Harris- and Hessian-Laplace detectors to obtain invariance against affine transformed images. This approach is based on the shape estimation properties of the second moment matrix of intensities. However, the simultaneous optimization of spatial point location, scale and shape comes at higher computational cost.

A similar idea to Harris-Laplace and Hessian-Laplace is used by Lowe (Lowe, 1999, 2004) in his *Difference of Gaussian* (DoG) detector. DoG approximates the scale normalized Laplacian by calculating differences of Gaussian blurred images at several and adjacent local scales. An accurate point of interest localization procedure, elimination of edge responses by Hessian based analysis and orientation assignment with orientation histograms completes the carefully designed detector.

Maximally Stable Extremal Regions (Matas et al., 2004) (MSER) is another famous detector based on intensity values and connected component analysis of an appropriately thresholded image. The obtained regions are of arbitrary shape and they are defined by the border pixels enclosing a region, where all the intensity values within the region are consistently lower or higher with respect to the surrounding. The main advantage of this detector is the fact, that the obtained regions are robust against continuous (an thus even projective) transformations and even non-linear, but monotonic photometric changes. In the case a single interest point is needed, it is usual to calculate the center of gravity and take this as an anchor point,

e.g., for obtaining reliable point correspondences. In contrast to the detectors mentioned before, the number of regions detected is rather small, but the repeatability outperforms the other detectors in most cases.

The detections of points and/or regions of interest is followed by the development of *feature descriptors*. Feature descriptors are used to describe the points/regions or its local neighborhood already identified by the detectors by certain invariance properties. Invariance means that the descriptors should be robust against various image variations such as affine distortions, scale and illumination changes or compression artifacts.

Locally Binary Patterns (LBP) is a very simple texture descriptor initially proposed in (Ojala et al., 1996). LBP is invariant to monotonic gray value transformations but it is not inherently rotational invariant. Nevertheless this can be achieved by rotating the regions of interest. Furthermore, scale invariance can be reached in combination with scale invariant detectors.

Shape context descriptors have been introduced by Belongie *et al.* (Belongie et al., 2002) in 2002. They exploit internal or external contour points of the investigated object or region. The contour points can be detected by any edge detector and are regularly sampled over the whole shape curve. A full shape representation can be obtained by taking into account all relative positions between two different contour points and their pairwise joint orientations. It is obvious that the dimensionality of such a descriptor heavily increases with the size of the region. To reduce the dimensionality a coarse histogram of the relative shape sample points coordinates is computed – the *shape context*.

Scale Invariant Feature Transform (SIFT) proposed by (Lowe, 1999, 2004) is one of the most popular descriptors with excellent performance (Mikolajczyk and Schmid, 2005). It combines a scale invariant detector (DoG) and a rotation invariant descriptor based on gradient orientation histograms. Although, SIFT is not affine invariant, it can be calculated on other type of detectors, so that it can inherit affine invariance from them (e.g. Harris-Laplace or MSER). Ke and Sukthankar in (Ke and Sukthankar, 2004) proposed a modified version of SIFT by reducing the dimensionality of the descriptor. Instead of gradient histograms on point of interest, they applied Principal Component Analysis to the scale-normalized gradient patches obtained by the DoG detector. This way the patch of local gradient orientations is described with the most significant eigenvectors. Although, following this approach the dimensionality of the descriptor can be reduced by a factor about eight, evaluations show that it performs slightly worse than standard SIFT descriptor (Mikolajczyk and Schmid, 2005).

Speed-Up Robust Features (SURF) was introduced in (Bay et al., 2006). As the name suggests, it is a speeded-up version of SIFT. The SIFT approach uses cascaded filters to detect scale-invariant characteristic points, where the DoG is calculated on rescaled images progressively. In SURF, square-shaped filters are used as an approximation of Gaussian smoothing. One big advantage of this approximation is that, convolution with square-shape filters can be easily calculated with the help of integral images. And it can be done in parallel for different scales. SURF descriptor is scale and rotation invariant and exploits Haar wavelet responses (again, use of integral images reduces computational time) in horizontal and vertical direction to describe points of interest. Analysis (Juan and Gwun, 2009) shows that SURF is three times faster than SIFT, while its performance is comparable to SIFT. Although,

SURF is good at handling images with blurring and rotation, it is not good at handling viewpoint and illumination changes.

Since the appearance of SIFT and SURF a huge variety of local descriptors has been proposed. However, the performances of SIFT and SURF on object recognition and image matching tasks are still used as benchmarks for every newly introduced descriptor.

2.1.2 *Global Properties of Objects*

In this section we discuss methods that are used to encode the global properties of objects. The main idea for all this methods is to project the original input images onto a suitable lower dimensional subspace that represents the data best for a specific task. By selecting different criteria for the projected data different methods can be derived.

Principal Component Analysis (PCA) (Jolliffe, 2002) also known as is a well known and widely used technique in statistics. It was first introduced by Pearson (Person, 1901) and was independently rediscovered by Hotelling (Hotelling, 1933). The main idea is to reduce the dimensionality of data while retaining as much information as possible. This is assured by a projection that maximizes the variance but minimizes the mean squared reconstruction error at the same time. PCA was introduced to Computer Vision by Kirby and Sirovich (Kirby and Sirovich, 1990) and became popular since Turk and Pentland (Turk and Pentland, 1991) applied it for face recognition. Non-negative matrix factorization (NMF) was introduced by Lee and Seung (Lee and Seung, 1999) in computer vision for object representation. In contrast to PCA, NMF does not allow negative entries whether in the basis nor in the encoding. As a result we obtain additive basis vectors mostly representing local structures. Thus, if the underlying data can be described by distinctive local information the representation may be sparse.

When the pose or the structure of the objects are of main interest, PCA and NMF can be proven very useful for encoding object properties. However, in many tasks, motion information is more important. In such cases PCA and NMF are useless, and different global descriptors should be developed capable to encode the presence of motion in a scene.

One of the most common approaches towards motion encoding is by using *background subtraction* techniques. Such techniques usually transform raw images to binary ones, whose zeros correspond to background (objects of non-interest) and ones to foreground (objects of interest). The task of background subtraction constitutes a key component for locating moving objects, facilitating search space reduction, modeling visual attention and, most important, relating objects to events.

Background subtraction techniques applied on video sequences model the color properties of depicted objects (Brutzer et al., 2011; Herrero and Bescos, 2009) and can be classified into three main categories (El Baf et al., 2009); basic background modeling (McFarlane and Schofield, 1995; Zheng et al., 2006), statistical background modeling (Elgammal et al., 2000; Wren et al., 1997) and background estimation (Messelodi et al., 2005; Toyama et al., 1999). The most used methods are the statistical ones due to their robustness to critical situations. For statistically representing the background, a probability distribution is used to model the history of pixel intensities over time.

Another common approach for encoding motion in a scene is called Motion History Images (MHI) (Davis, 2001b). The MHI is a static image tem-

plate where pixel intensity is a function of the recency of motion in a sequence. MHI automatically performs temporal segmentation, is invariant to linear changes in speed, and due to its low computational cost can run in real-time on a standard platforms.

2.2 THE NOTION OF EVENT

Literally, an event is an occurrence happening at a determinable time and place, with or without the participation of human agents and it may be a part of a chain of occurrences as an effect of a preceding occurrence and as the cause of a succeeding occurrence. In other words, an event is a semantically meaningful activity taking place within a selected environment and containing a number of necessary objects.

Therefore, events are inherently related to objects. In computer vision society, Zelnik-Manor and Irani in (Zelnik-Manor and Irani, 2001) define events as long term temporal objects, which are characterized by spatio-temporal features at multiple levels. Furthermore, Polana and Nelson (Polana and Nelson, 1994) separate events into three different classes; i) *temporal textures*, which are of indefinite spatial and temporal extent, ii) *activities*, which are temporally periodic but spatially restricted and iii) *motion events*, which are isolated events that do not repeat either in space or in time.

Based on the above, events should be described by their *behavioral content* in order to be isolated and discriminated within long continuous video sequences. By term behavioral content we refer to spatial and temporal features of events. Although, spatial features may appear at different scales in different images (see section 2.1), due to the perspective nature of the projection in the spatial dimension, temporal features are always characterized by the same temporal scale. For example, a single step of a walking person, viewed by two different video cameras of the same frame rate, will extend over the same number of frames in both sequences, regardless of the internal or external camera parameters.

Elegant methods have been proposed for analyzing events and capturing their spatial and temporal characteristics by specialized models (Black and Yacoob, 1997; Cutler and Davis, 2000). Construction of these models is usually done via an extensive learning phase, where many examples of each studied action are provided and by exploiting prior knowledge about the types of events, their temporal and spatial extent and/or their nature (periodic/non periodic).

In the following we present three different real-world use cases, where we exploit information about objects as a basis for understanding events. Specifically, the first use case describes the development of a supportive vision based system for detecting in real-time elderly and/or patients fall incidents in indoor environments. In the second use case a maritime security vision based system is proposed, while in the third use case we present a surveillance system for activity recognition in industrial workflow. Initially, we present the motivation for each one of the use cases and then, following the David Marr's *levels of understanding* framework, via literature review we formulate in detail the problem, along with its constraints and requirements, we present a solution design and implementation and, finally, we evaluate the performance of the proposed solution.

3 VISION BASED FALL DETECTION SYSTEM

3.1 MOTIVATION

According to demographic and epidemiological data, the number of people over 65 years old is increasing six times faster than the rest population on earth. Indicatively, about one in eight Americans were elderly in 2000, but about one in five would be elderly by the year 2050 (Shrestha and Heisler, 2011), (Fallwatch, 2009). People's ability to live independently with dignity, without having the need to be attached to any person in order to live a normal life and fulfill daily living activities, affects greatly their quality of life. However, emergency department visits, related to falls, are more common in children less than five years old and adults over 65 years old. Compared to children, elderly who fall are ten times more likely to be hospitalized and eight times more likely to die (Runge, 1993), (Fallwatch, 2009). Thus, falls can be considered as one of the most important problems that hinder these people ability to live an independent life.

In order to understand the elderly fall problem, and try to prevent fall incidents, someone needs to examine where they occur. Recent studies show that 67% of fall incidents take place inside or in close proximity to patients' home and residential institutions where a medical alert system can be of immediate assistance. Taking into consideration the importance of humans' fall problem and the aforementioned statistics, the development of robust home surveillance systems is necessary. For this reason, a major research effort has been conducted in the recent years for automatically detecting persons' falls.

One common way for automatic fall detection is through the use of specialized devices, such as accelerometers, floor vibration sensors, barometric pressure sensors, gyroscopic sensors, or combination/fusion of them (Wang et al., 2005), (Nyan et al., 2008), (Bianchi et al., 2010), (Zigel et al., 2009) and (Le and Pan, 2009) or help buttons (Alert, 2015). However, most of these approaches require specific wearable devices that should be attached to patients' body, and thus, their efficiency relies on the persons' ability and willingness to wear them. External sensors, such us floor vibration detectors, require a complex setup and are still in their infancy, while in the case of a help button, it is useless, if the person is unconscious after the fall.

A more challenging alternative is the use of visual sensors, which is however a prime research issue due to the complexity of visual content, i.e. illumination variations, background changes and occlusions, and the fact that a fall incident should be discriminated over other ordinary humans' activities. The emergence of computer vision systems has allowed researchers to overcome the aforementioned problems; vision based systems are less intrusive, can be installed on buildings and are not worn by users.

Furthermore, cameras can provide a vast amount of information about patients and environment making vision based systems suitable for different kind of applications, as they are able to detect several events simultaneously. For example, a vision based system can be used to detect fall incidents, and at the same time, to check other daily life activities, like medication intake.

Apart form the fact that vision based systems can provide valuable information towards falls detection, at the same time, they can preserve persons' privacy by exploiting an event-based design that triggers alarms and/or enables video recording only after the occurrence of specific predefined events.

In the following we present computer vision based approaches towards the fall detection problem. A detailed survey of fall detection methodologies is presented in (Noury et al., 2007).

3.2 COMPUTER VISION RELATED WORKS

Vision based fall detection systems can be divided into three categories according to the number of cameras they use and the nature of visual sensors – visual spectrum sensors and depth sensors. This way, there are *monocular camera*, *multi-camera* and *depth camera* fall detection systems.

3.2.1 *Monocular Camera Systems*

Some works that exploit 2D image data in order to detect fall incidents or to recognize humans' activities, including falls, are (Doulamis, 2010), (Fu et al., 2008), (Doulamis and Makantasis, 2011), (Foroughi et al., 2008b), (Foroughi et al., 2008a), (Makantasis et al., 2012) and (Debard et al., 2012). These approaches use image segmentation algorithms to extract the foreground objects, in order to exploit their shape and produce features suitable for fall incidents discrimination. For example, in (Rougier et al., 2011b) a method that uses a shape matching technique to track the person's silhouette along the video sequence, is presented. The shape deformation is then quantified from these silhouettes based on shape analysis methods and falls are detected from normal activities using a Gaussian mixture model.

The most commonly used features are foreground object's projected height-width ratio, as well as, its centroid and/or its head 2D vertical motion velocity, which besides fall discrimination can be describe how severe a fall can be. Methods based on projected height-width ratio estimation are efficient, when the camera is placed sideways and at the height of foreground object's center of mass. However, such camera mountings result in fall detection systems sensitive to objects occlusions. For more realistic situations, the camera has to be placed higher in the room to avoid occluding objects and to have a larger field of view. Moreover, 2D velocity has greater value when a person is near the camera and smaller value when he/she steps away from it. Consequently, threshold values to discriminate falls than other ordinary activities can be difficult to define.

The aforementioned approaches are inherently depended on the efficiency of image segmentation algorithms and assume that all moving objects, correspond to foreground objects, are humans. The authors of (Qian et al., 2008) overcome this problem by presenting a more sophisticated approach based on human anatomy. They assume that each part of the human body occupies an almost fixed percentage in length relative to total body height. Based on this assumption, they train a classifier capable to discriminate six indoor human activities, including fall incidents.

Although, the aforementioned works present good results, none of these exploits three dimensional information to increase system's robustness. Contrary to these approaches, our system overcomes the aforementioned problems by extracting and using real-world three dimensional information, de-

spite the fact that it uses a single monocular camera. Concretely, it extracts and uses the actual height and width of foreground objects, which are view invariant features and provide additional information, sufficient for discriminating, if the moving object is a human or something else (e.g. a cat or a dog). In addition, vertical motion velocity is represented by the time derivative of actual height of a foreground object, and thus is not affected by the distance between a person and the camera.

3.2.2 *Multi-Camera Systems*

Multi-camera systems have been also proposed for detecting fall incidents. In the work of (Thome et al., 2008), motion is modelled by a HMM. The features, used for motion analysis, are extracted from a metric image rectification. Posture classification is performed by a fusion unit, merging the decision provided by the independently processing cameras in a fuzzy logic context. Anderson *et al.* (Anderson et al., 2009) represent a person by a three-dimensional voxel, called voxel person. They recognize humans' activities from linguistic summarizations of temporal fuzzy inference curves representing the voxel person's states and detect fall incidents by using a hierarchy of fuzzy logic.

Auvinet *et al.* (Auvinet et al., 2011) use multiple cameras to reconstruct the 3D shape of people and fall incidents are detected by analyzing the shape's volume distribution along the vertical axis. A fall alarm is triggered when the major part of this distribution is abnormally near to the floor during a predefined period of time. Hazelhoff *et al.* (Hazelhoff et al., 2008), proposed a multiple camera system, which detect falls by using the direction of the principal component and the variance ratio of the human silhouette. To further reduce false alarms, it exploits vertical motion velocity of foreground object's head.

Nevertheless, an important point about multi-camera systems is that they require calibrated cameras, synchronized video sequences and employment of computational demanding stereo-vision mathematics in order to extract reliable, three dimensional information. The presented system extracts three dimensional features by using a single monocular calibrated camera and thus overcomes the problem of video stream synchronization and increased computational cost due to stereo-vision. Moreover, it uses a self calibration technique to further reduce installation cost. Furthermore, our system computes three dimensional features without the need of fusing different camera streams, which increases computational cost.

3.2.3 *Depth Camera Systems*

The use of depth cameras, based on Time-of-Flight technology, is another convenient way to extract 3D measures to detect fall incidents by using a single device. In the works of (Diraco et al., 2010), (Grassi et al., 2010) and (Rougier et al., 2011a) depth cameras are used to extract 3D shape of foreground object and recognize fall incidents based on human centroid height relative to the ground plane and body velocity. Mastorakis and Makris in (Mastorakis and Makris, 2012) use a 3D box that bounds the foreground object to measure its vertical velocity. Fall incidents detection is based on the value of vertical velocity and inactivity duration. Dubey *et al.* in (Dubey et al., 2012) use a depth camera to create 3D Motion History Images that

contain three channels. For each channel the seven hu-moments are calculated and the 21 extracted features are used as input to an SVM in order to discriminate falls than other activities.

Despite the fact that these approaches take into consideration three dimensional information, depth cameras use short range sensors, which may cause various problems. They don't take into account the orientation of the moving blob, and measures that are provided could be affected by reflectivity objects properties and aliasing effects when the camera-target distance overcomes the non-ambiguity range.

3.3 APPROACH OVERVIEW

To successfully design a fall detection system, on the one hand we have to investigate what are the falls characteristics and what features can discriminate a fall than other human activities, and on the other, we have to define fall detection process steps and identify and overcome the difficulties associated with them.

3.3.1 *Falls Characteristics*

As it is mentioned in (Noury et al., 2007), fall incidents can be characterized by sudden and high speed vertical motion, rapid human posture changes and, usually, they are followed by lack of significant movement.

3.3.1.1 *Vertical Motion Velocity*

It is one of the most commonly used motion features to detect fall incidents and besides falls discrimination it provides useful information about the fall intensity and thus possible injuries. Vertical motion velocity can be defined as the time derivative of human height

$$V = \nabla h_a(t) \tag{3.1}$$

where $h_a(t)$ stands for the actual height of a human in 3D space at time instance t.

3.3.1.2 *Human Posture Changes*

In contrast to ordinary human activities, during which human posture is changing slowly, during a fall, human posture is changing suddenly. On the one hand, human posture can be characterized by person's width-height ratio, and this is valid as this ratio is bigger in value when a fall event occurs than the same ratio with the person in standing position, and on the other by the orientation of person's body (Foroughi et al., 2008a).

3.3.1.3 *Lack of Significant Movement*

This feature used to describe how severe a fall can be. It based on the assumption that after a serious fall incident, the person will stay immobile, at least for some time. This feature is not adequate to discriminate a fall. Although it can be used along with the aforementioned features to decrease false positive alarms, it may increase false negative rate.

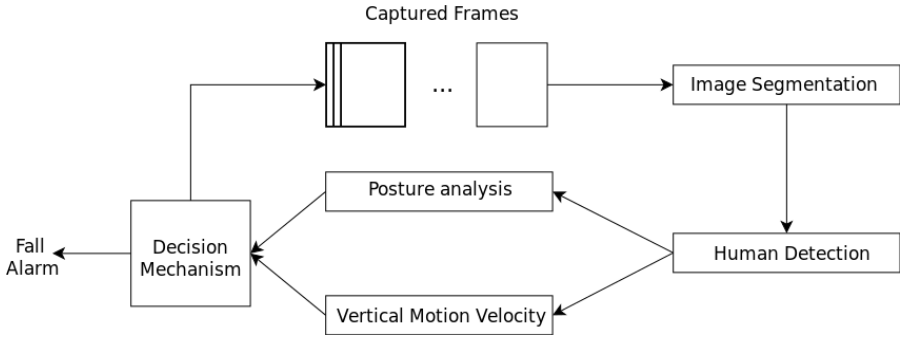


Figure 3.1: Proposed fall detection system architecture.

3.3.2 *System Architecture*

The proposed fall detection mechanism includes tree phases: a) foreground/ background extraction and human detection, b) appropriate feature extraction and c) the decision mechanism utilization. System overall architecture is presented in Figure 3.1.

The first step of our algorithm is to detect the persons in a scene for every captured frame. This can be done by applying image segmentation and background subtraction methods. Once the persons have been detected, the system tracks them and estimates their height in order to calculate vertical motion velocity and to analyze their posture. Finally, a decision mechanism uses the aforementioned features, in order to detect falls and trigger the alarm.

In our approach, we chose to combine the first two features of subsection 3.3.1, in order to detect fall incidents. The use of the third feature would result in decreasing false positive rates and increasing the false negative ones, due to the fact that lack of significant movement is not necessarily occurred after a fall incident. However, the primary concern of a fall detection system is to achieve low false negative rates, i.e. its goal is to detect all fall incidents even if some of them are not true (false alarms). Thus, we prefer to exclude the third feature from our analysis.

3.3.3 *Visual Constraints and Challenges*

In this work, we focus on two different things. Firstly, we want to develop a robust and low-cost system, which will use simple low-end web or IP cameras. Secondly, this system has to be capable to operate in real-time. Given the equipment constraints and the nature of the problem and by knowing that our system's performance is inherently depended on the efficiency of image segmentation, as it is the first step during fall detection process, we deal with the following challenges:

1. Segmentation algorithms have to be capable to handle cluttered and textured background.
2. The system has to operate properly in dynamically changing visual conditions, such as variable illumination.
3. Falls can happen in any direction according to the camera position and have to be detected, even if humans are partially occluded.

4. Decision mechanism must be robust to discriminate falls from other ordinary human activities that may look like falls, but they are not, such as sitting and bending.
5. Falls are sudden events, so the overall fall detection process has to be characterized by low computational cost and memory requirements, in order to operate in real-time.

In the following we describe step by step the fall detection process and we will show how we handle these challenges.

3.4 IMAGE SEGMENTATION

The term image segmentation algorithms refer to background subtraction and foreground extraction techniques. The requirements of image segmentation techniques, for the fall detection system, are defined knowing that 67% of humans' fall incidents take place inside or in close proximity to patient's home. These requirements can be divided into two categories in regard to the operation mode as: *indoor environment operation requirements* and *outdoor environment operation requirements*.

There are some differences between indoor and outdoor environment operation. Illumination conditions can dramatically and suddenly change in indoor environments, due to artificial light sources, while in outdoor environments illumination, usually, changes progressively and slow. Furthermore, outdoor environments present more complicated background with higher background motion and more moving objects compared to indoor environments.

Our approach exploits the best possible components to adapt to the needs of operational requirements. In other words, different segmentation techniques are used for indoor and outdoor environments. The term indoor environment corresponds to patients' homes, clinics, elderly nursing homes, while outdoor environment refers to areas in close proximity to patients' home (e.g. backyard, garden). Thus, three segmentation techniques were evaluated. Each of them uses a different approach to segmentation problem.

We test the following algorithms: (i) *Iterative Scene Learning* algorithm (ISL) presented in (Doulamis, 2010), (ii) *Adaptive Student-t Mixture Model* background subtraction (ASMM), presented in (Makantasis et al., 2012) and (iii) *non-Parametric Background Generation* (nPBG), presented in (Liu et al., 2007). This choice is justified by the fact that ISL algorithm extracts the foreground by using motion information in the scene, ASMM subtracts the background by using a parametric approach to learn background pixels intensities and nPBG learns the same intensities in a non-parametric way.

3.4.1 *Iterative Scene Learning algorithm*

ISL is a light-weight, foreground extraction algorithm capable to operate in real-time and in complex, dynamic in terms of visual content, and unexpected environments. It uses the "pyramidal" Lucas-Kanade algorithm (Lucas and Kanade, 1981) to estimate and exploit the intensity of motion vectors along with their directions in order to identify foreground objects' movements. Then, it estimates high motion information areas by using a

binary mask which was obtained by thresholding the absolute difference between two subsequent frames.

ISL has the ability to catch large motions by using an image pyramid overcoming this way the problems that arise by the assumptions of brightness constancy, temporal persistence and spatial coherence. To surpass the difficulties caused by the sensitivity of motion vectors to luminosity changes and camera parameters, the methodology of (Shi and Tomasi, 1994) was applied to detect salient points on video frames that are considered as "good features" for estimating motion vectors. The feature points are extracted by constraining the high motion information areas on the aforementioned binary mask in previous frames. Then, the initially detected feature points are spatially sampled by retaining the local maximum feature points within a neighboring region.

The aforementioned procedure indicates the high motion information areas of a scene. Then, motion information is used as a computationally efficient background/foreground updating mechanism that updates the background at every frame instance. In particular, regions which are spatially far away from the motion activity segments are denoted as background areas. Based on estimated background area, subtraction techniques provide estimates of the foreground objects.

More specifically, when background updating takes place, the regions that are far away from the estimated high motion information areas are denoted as background while the ones that are close to high motion information areas are considered as ambiguous regions. If motion vectors intensity in ambiguous regions are greater than a threshold the background values are being updated since it is assumed that a foreground object has appeared and covered its parts. Otherwise there is no important variation in the scene which imposes that there is no need for background updating.

3.4.2 *Adaptive Student's-t Mixture Model*

Adaptive Student's-t Mixture Model (ASMM) was inspired by Gaussian Mixture Model (Stauffer and Grimson, 1999) (GMM). We choose to use ASMM instead of GMM since: (i) Student's t-distribution presents more concentrated form and, as explained in (Chatzis and Varvarigou, 2009), it has been proposed as an alternative to GMM to resolve problems related with outliers, thus making it more robust to artifacts such as shadows, and (ii) only two factors, pixel value representation and distribution degrees of freedom (DoF), are needed during the modeling process.

ASMM considers that each image pixel can have two states, background or foreground. In the beginning, this algorithm subtracts the current frame for processing by the previous one, in order to find candidate foreground pixels (pixels with different intensity between two subsequent frames) and the background modeling technique is applied only on these points. By this approach for every captured frame there is no need for processing all image pixels but only a subset of them, reducing this way computational cost. It has to be mentioned that this algorithm reduces further the computational cost by working with gray-scale images and using univariate Student-t probability density function, instead of multivariate.

In details, at any time what is known about a particular pixel, (x_0, y_0), is its series of values, $\{X_1, ..., X_k\} = \{I(x_0, y_0, i) : 1 \leq i \leq k\}$, where I is the image sequence. This time series of pixel values can be modeled by K

Student's-t distribution mixtures. K is determined by the available memory and computational power and can be defined between 3 and 5. Each new pixel value, X_n, is modeled by these mixtures with a probability

$$P(X_n) = \arg\max_{\theta} p_\theta(x_\theta) \ , \tag{3.2}$$

where $p_\theta(x_\theta)$ is

$$p_\theta(x_\theta) = \frac{\Gamma(\frac{v_\theta+1}{2})}{\sqrt{v_\theta \pi}\,\Gamma(\frac{v_\theta}{2})}(1+\frac{x_\theta^2}{v_\theta})^{-\frac{v_\theta+1}{2}} \ . \tag{3.3}$$

$\theta = [1, 2, ..., K]$, v_θ stands for the degrees of freedom of θ^{th} mixture and x_θ is the absolute difference between X_θ (modeled value of θ^{th} mixture) and X_n. If $P(X_n)$ is bigger than a threshold for i^{th} mixture then X_n is modeled by i^{th} mixture and this pixel is denoted as background. The degrees of freedom for every mixture are updated using

$$v_\theta = \begin{cases} v_\theta - 1 & \text{if } \theta = i \text{ and } v_\theta > 1 \\ v_\theta + 1 & \text{if } \theta \neq i \text{ and } v_\theta < \beta \end{cases} \tag{3.4}$$

and modeled pixel value X_θ is updated as

$$X_\theta = \begin{cases} (1-\lambda)X_\theta + \lambda X_n & \text{if } \theta = i \\ X_\theta & \text{if } \theta \neq i \end{cases} \ . \tag{3.5}$$

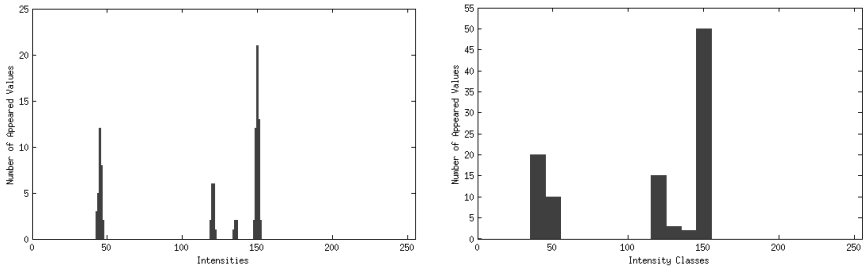
In Eq.(3.4) β is the maximum allowed number of degrees of freedom for a mixture. In Eq.(3.5) $\lambda = p_\theta(x_\theta)/p_\theta(0)$.

Eq.(3.4) is used to update the importance of mixtures in modeling process, while Eq.(3.5) updates the modeled pixel value of mixtures. The importance of mixtures can be described by their degrees of freedom; a mixture whose degrees of freedom are decreased is able to model a wider range of pixel values, and thus its importance is increased.

If none of the mixtures is capable to model a new pixel value X_n, then the degrees of freedom of each mixture are increased by one and the importance of the less probable mixture is checked. If this mixture has more degrees of freedom than a threshold β, then this mixture is replaced by a new mixture whose DoF is the average of maximum and minimum DoF of K distributions that model the specific pixel. The modeled pixel value for the new component is set to X_n, and, finally, the corresponding pixel is denoted as foreground.

3.4.3 *Non-Parametric Background Generation*

Non-Parametric Background Generation algorithm, in the beginning, converts image color space to YUV and uses Y channel to model, pixel-wise, scene intensities. For initialization it uses a history of intensity values for each pixel. Then, it creates the histogram for each pixel's history, uses Mean-Shift algorithm to find histogram's modes and it selects a predefined number of dominant modes (maxima with most appearing intensity values), as the most reliable background modes. Each new pixel value is checked against the pixel's most reliable background modes and if their difference is smaller than a predefined threshold the pixel is denoted as background. Otherwise, the pixel is denoted as foreground, pixel's history is updated to include the new value and most reliable modes are re-calculated.



(a) Original histogram (b) Quantized histogram

Figure 3.2: (a) Histogram for each intensity value and (b) histogram divided into intensity classes.

We followed this approach, but instead of using computational expensive Mean-Shift algorithm, we divide each histogram to a predefined number of classes. Each class covers an intensity range and contains the sum of appeared pixel intensities between its range. This process is shown in Figure 3.2. Following this approach the most reliable background modes are represented by the largest size classes.

3.5 FEATURES EXTRACTION FOR FALL DETECTION

In this section, we provide a description for both 2D features and 3D features, utilized by the proposed methodology. 2D features include person's projected width-height ratio and person's body orientation, while 3D features include vertical motion velocity based on person's actual height estimation.

All features are extracted using a simple monocular camera. Generally, there are no limitations regarding the maximum distance of the camera from the falling scene, neither for the resolution, as long as the human spans an area of more than 40 pixels in each frame, in order to achieve robust segmentation of foreground objects.

3.5.1 *2D Features*

Width-height ratio is determined by person's projected width and height. So, the first step for the computation of this ratio is the estimation of these two measures. Both of these measures can be estimated by the four corners of a minimum bounding box that includes the person. By using the four corners of the minimum bounding box the points q_{bm}, q_{tm}, q_{lm} and q_{rm}, that correspond to foreground object's bottom-most, top-most, left-most and right-most points, can be obtained. By using these four points, width-height ratio can be expressed by Eq.(3.6).

$$R = \frac{w_p}{h_p} = \frac{q_{rm} - q_{lm}}{q_{tm} - q_{bm}} \ , \tag{3.6}$$

where w_p and h_p stand for the projected width and height of the foreground object.

Orientation of a person's body can be successfully described by the orientation of an ellipse that best bounds the person. The approximation of such an ellipse requires to define its center $(\bar{x}, \bar{y})$, its orientation, which is

the angle ϕ of its major semi-axis and the lengths a and b of its major and minor semi-axes.

As described in (Foroughi et al., 2008a), a bounding ellipse can be approximated by image moments. Having estimate the bounding box that contains a foreground object, as shown in (Spiliotis and Mertzios, 1998) the computational cost for computing image moments is linear to the number of foreground object pixels. Thus, the complexity is independent on the size of the image, depending only on the size of the foreground objects. For an image with scalar pixel intensities $I(x,y)$, spatial image moments are given by Eq.(3.7).

$$M_{ij} = \sum_x \sum_y x^i y^j I(x,y) \quad for\ i,j = 0,1,2,\dots \ . \tag{3.7}$$

The center of mass of a person's body coincides with the center of the ellipse, and thus it can be obtained by

$$(\bar{x},\bar{y}) = (M_{10}/M_{00}, M_{01}/M_{00}) \ . \tag{3.8}$$

After the estimation of ellipse's center, central moments of second order can be used to estimate ellipse's orientation ϕ. Central moments can be computed by

$$\mu_{ij} = \sum_x \sum_y (x-\bar{x})^i (y-\bar{y})^j I(x,y) \quad for\ i,j = 0,1,2,\dots \tag{3.9}$$

and orientation ϕ by

$$\phi = \frac{1}{2} \arctan\left(\frac{2\mu_{11}}{\mu_{20}-\mu_{02}}\right) \ . \tag{3.10}$$

Finally, the lengths a and b of ellipse's major and minor semi-axes can be obtained by Eq.(3.11) by using the greatest and least, moments of inertia (Sobottka and Pitas, 1996).

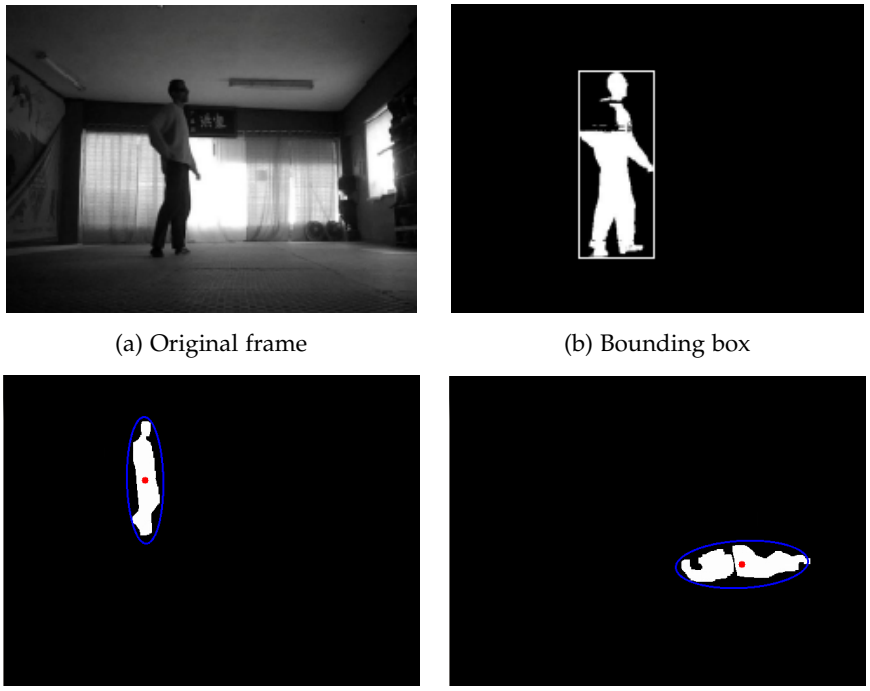
$$a = \left(\frac{4}{\pi}\right)^{\frac{1}{4}} \left[\frac{{I_{max}}^3}{I_{min}}\right]^{\frac{1}{8}} \ , \quad b = \left(\frac{4}{\pi}\right)^{\frac{1}{4}} \left[\frac{{I_{min}}^3}{I_{max}}\right]^{\frac{1}{8}} \ , \tag{3.11}$$

where

$$\begin{aligned} I_{max} &= \frac{\mu_{20}+\mu_{02}+\sqrt{(\mu_{20}-\mu_{02})^2+4\mu_{11}^2}}{2} \\ I_{min} &= \frac{\mu_{20}+\mu_{02}-\sqrt{(\mu_{20}-\mu_{02})^2+4\mu_{11}^2}}{2} \end{aligned} \tag{3.12}$$

correspond to greatest and least moments of inertia.

In Figure 3.3, 2D features extraction is presented. Figure 3.3(a) and Figure 3.3(b) show the original captured frame along with the extracted foreground and minimum bounding box creation. This minimum bounding box, as mentioned before, is used to estimate person's width-height ratio. Figure 3.3(c) and Figure 3.3(d) show the estimated bounding ellipse for two different human positions, standing position and after a fall incident respectively.



(a) Original frame (b) Bounding box

(c) Approximated ellipse (d) Approximated ellipse

Figure 3.3: (a) Original frame, (b) minimum bounding box creation to extract width-height ratio, (c) approximated ellipse for standing position and (d) approximated ellipse after a fall incident.

3.5.2 *3D Features*

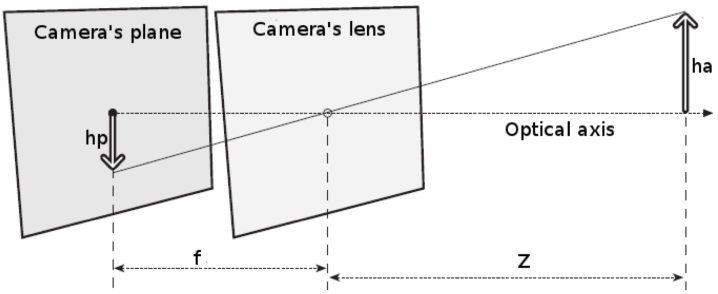
3D features include the vertical motion velocity estimation based on the actual person's height. We choose to estimate vertical motion velocity by using the actual person's height, due to the fact that actual height is a view independent measure and thus vertical motion velocity is not affected neither by the viewpoint nor by the distance between the camera and the person.

By using a pinhole camera an object with actual height h_a is represented on camera's plane with projected height h_p, as shown in Figure 3.4(a). By examining this representation, it appears that the actual height of the object is given by

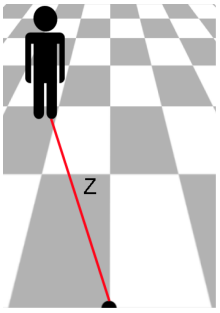
$$h_a = Z\frac{h_p}{f} \tag{3.13}$$

if camera's focal length, f, distance, Z, between the camera and the object and object's projected height, h_p, are known.

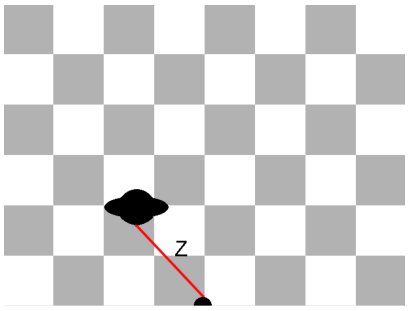
As explained in subsection 3.5.1, object's projected height is already known. So, the problem of estimating object's actual height includes, firstly, the transformation of our camera to a pinhole-like camera, with known focal length and distortion-free capturing, and, secondly, the estimation of the distance between the camera and the object. The first problem is addressed by using camera calibration techniques, while we overcome the second one by the construction of a reference plane that is the orthographic view of the floor, as shown in Figure 3.4(c). On the reference plane the relation between camera's natural units (pixels) and the units of the physical world



(a) Pinhole camera model



(b) Camera plane



(c) Reference plane

Figure 3.4: (a) pinhole camera-depiction of an object with actual height h_a to camera's plane with projected height h_p, (b) camera's plane, (c) reference plane – the distance between the camera and the person is linear to the number of pixels.

(cm) is linear and thus the distance between the camera and the person can be straightforward calculated. At this point it has to be mentioned that we use a self calibration technique to reduce installation cost.

3.5.2.1 *Camera Self-Calibration*

Camera calibration is a necessary process, in order to obtain camera's focal length, which is required for actual foreground object's height approximation. The self calibration technique that we chose to use requires three finite vanishing points that correspond to three mutually orthogonal planes of the scene, in order to compute camera's geometry (principal point and focal length). Our system uses a single stationary monocular camera, and thus only a single view of the scene is available. In order to detect three finite vanishing points that correspond to three mutually orthogonal planes of the scene the camera has to be placed in a way that its plane is not parallel to any of the aforementioned planes.

During self-calibration, for vanishing points detection, the unbounded image plane is chosen as accumulator space since it preserves the original distances among points and lines (Grammatikopoulos et al., 2007). The intersections of all pairs of line segments are selected as accumulator cells. These accumulator cells represent potential vanishing points. Since a vanishing point in a 3D scene is a point at infinity, the corresponding vanishing point in 2D image cannot lie on a line segment. So, from the initial set of candidate vanishing points all of them that do not satisfy the aforementioned constraint are removed. For each one of the candidate vanishing

points the contribution of every line segment is computed by means of a voting scheme.

Next, all candidate vanishing points along with their corresponding scores are checked against certain geometrical criteria (Rother, 2002), including orthogonality criterion and camera criterion. Every triplet of finite vanishing points forms a triangle. The intersection point of its heights defines the principal point and its size defines the focal length. According to the orthogonality criterion each angle of this triangle has to be smaller than 90°. The camera criterion is fulfilled if the principal point and the focal length are inside a certain range. So, only triplets of finite vanishing points that form acute triangles and present "reasonable" values for principal point and focal length are considered. These triplets are sorted according to their total score; that with the highest score is chosen as the final triplet of dominant vanishing points.

To estimate principal point (x_o, y_o) and camera's focal length f, each pair of orthogonal vanishing points, $\mathbf{v_1}$ and $\mathbf{v_2}$, expressed in homogeneous coordinates, supplies a linear constraint on the entities of conic c of the form

$$\mathbf{v_1}^{\mathrm{T}} c \mathbf{v_2} = 0 \ . \tag{3.14}$$

By ignoring image aspect ratio and skewness, c may be written as

$$c = \begin{bmatrix} 1 & 0 & -x_o \\ 0 & 1 & -y_o \\ -x_o & -y_o & {x_o}^2 + {y_o}^2 + f^2 \end{bmatrix} \ . \tag{3.15}$$

After focal length estimation, captured frame should be rectified by removing radial lens distortions. Radial lens distortion at any image point (x_d, y_d) can be modeled by the first two coefficients of a Taylor series around $r = 0$, where r is the distance between point (x_d, y_d) and principal point (x_o, y_o). Radial lens distortion is given by

$$\begin{aligned} x_u &= x_d + x_d(k_1 r^2 + k_2 r^4) \\ y_u &= y_d + y_d(k_1 r^2 + k_2 r^4) \end{aligned} \ , \tag{3.16}$$

where (x_u, y_u) is the undistorted point corresponding to distorted point (x_d, y_d). To detect lines we used a straight line detector with increased tolerance region, so as segments of curved lines are detected as straight lines (Thormaehlen et al., 2003). Then, the detected lines are constrained to converge to their corresponding vanishing point (x_v, y_v) according to the following equation

$$(x - x_v)cos\omega + (y - y_v)sin\omega = 0 \ , \tag{3.17}$$

where (x, y) are the image coordinates of an individual point on a line and ω is the angle between the line and the vertical image axis. By introducing the coefficients, k_1 and k_2, of radial distortion Eq.(3.17) results in (Grammatikopoulos et al., 2007):

$$\begin{aligned} & [x - (x - x_o)(k_1 r^2 + k_2 r^4) - x_v]cos\omega + \\ + \quad & [y - (y - y_o)(k_1 r^2 + k_2 r^4) - y_v]sin\omega = 0 \ , \end{aligned} \tag{3.18}$$

where (x_o, y_o) are the coordinates of the principal point. Coefficients k_1 and k_2 are computed so as the root mean square distance of points (x, y) from the fitted line is minimized.

3.5.2.2 *Reference Plane Construction*

A reference plane that represents the orthographic view of the floor, can be constructed by applying perspective transformations on the original captured frame. As described in (Cyganek, 2007), for a projective space $\wp^n$ a projective homography is defined as a nonsingular matrix $\mathbf{H}_{(n+1)\times(n+1)}$ with elements belonging to an affine space $\Re^n$, and defined up to a certain scalar value, called a scaling coefficient. A point $\mathbf{x}$ is projectively transformed to $\hat{\mathbf{x}}$ by

$$\hat{\mathbf{x}} = \mathbf{H}\mathbf{x}, \quad \mathbf{x}, \hat{\mathbf{x}} \in \wp^n \ , \tag{3.19}$$

where $\mathbf{H}$ is the coordinate transformation matrix (homography matrix).

In perspective transformations that are a specific case of projective homographies, called planar homographies, Eq.(3.19) can be expressed as

$$\begin{bmatrix} \hat{x}_1 \\ \hat{x}_2 \\ \hat{x}_3 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \ , \tag{3.20}$$

where $\mathbf{x}$ denotes pixel homogeneous coordinates in captured frame and $\hat{\mathbf{x}}$ is the new position of a pixel in the wrapped output image. By using perspective transformations any parallelogram can be transformed to any trapezoid, and vice versa. In our case, we want to transform the camera's plane to a reference plane that represents the orthographic view from above of the camera's plane. Then according to (Bevilacqua et al., 2008), $\hat{\mathbf{x}}$ and $\mathbf{x}$ can be expressed by the following relations

$$\hat{\mathbf{x}} = [\hat{x}\,\hat{y}\,1] \ \text{ and } \ \mathbf{x} = [x\,y\,1] \ , \tag{3.21}$$

where $x, y, \hat{x}, \hat{y}$ represent Cartesian coordinates on image plane and reference plane respectively and homography matrix $\mathbf{H} = [h_{ij}]$ can be normalized to have $h_{33} = 1$.

The algorithm of (Bevilacqua et al., 2008) finds the inverse perspective transformation that maps a set of three dimensional points (markers) of the real world object to the corresponding set of two dimensional points of a virtual grid, representing objects orthographic view from above. This algorithm uses a single image with a known pattern to extract a set of markers. To solve Eq.(3.20) at least four non collinear markers is required to be extracted, but usually a larger set of markers is available and the solution can be found in a least square method. Finally, the quality of the transformation is measured by computing the back projection error, E, associated with the homography matrix $\mathbf{H}$. This error is given by

$$\begin{aligned} E = \ & \sum_{i=1}^{n} \ (\hat{x}_i - \frac{h_{11}x_i + h_{12}y_i + h_{13}}{h_{31}x_i + h_{32}y_i + h_{33}})^2 + \\ & + \ (\hat{y}_i - \frac{h_{21}x_i + h_{22}y_i + h_{23}}{h_{31}x_i + h_{32}y_i + h_{33}})^2 \ , \end{aligned} \tag{3.22}$$

where n stands for the number of selected markers.

In our case, this algorithm has to be applied only once for a specific angle between camera's plane and floor plane. For a variety of angles, in order to reduce installation cost, this algorithm can be applied many times offline and its pre-calculated output can be used directly by the fall detection system.

3.5.2.3 *Actual Height Approximation*

As mentioned before, person's actual height can be approximated by Eq.(3.13) as shown in Figure 3.4(a). In order to use Eq.(3.13), we have to approximate the distance Z between the camera and the foreground object. This distance can be approximated by using the bottom-most point, q_{bm}, of foreground object. On the constructed reference plane the relation between camera's natural units (pixels) and the units of the physical world (cm) is linear and thus Z is straightforward calculated, Figure 3.4(c), while the other parameters of Eq.(3.13), focal length, f, and projected height, h_p are already known.

However, the actual height estimation can be affected by the motion of foreground object, as well as, by the appearance of errors during perspective transformations, as it depends on distance estimation on reference plane. Let us denote as $h(i)$ the approximated actual height of foreground object at the current frame of analysis i. In our approach, to reduce accumulation of approximation errors at the following frames for processing we use a heuristic iterative methodology, which updates the foreground height taking into account previous height information and the current one, $h(i)$. This methodology yields to a robust approximate solution, which is computed by

$$h(i) = \kappa h(i-1) + (1-\kappa)h(i) \ , \tag{3.23}$$

where κ is a parameter that regulates the importance of $h(i)$ to the iterative procedure. This iterative procedure requires an initial value of $h(i)$ which in our case is set to average height of adult males, e.g. 175cm. This initial value is not restrictive and can be set to any value according to the average height of potential foreground objects.

After, person's actual height approximation, vertical motion velocity can be estimated by Eq.(3.24), which, for a time window of length m frames, can be approximated by

$$V = \sum_{i=k-m}^{k} h_a(i) - h_a(i-1) \ , \tag{3.24}$$

where $h_a(i)$ stands for the actual height of a human in 3D space at the i^{th} image frame. Vertical motion velocity is calculated for a sequence m of frames and is an estimation of the speed of the motion and also an evidence of how severe a fall can be. Using person's actual height, measured in physical world units (e.g., cm, inches), (i) yields a more robust performance not affected by cases where the person is far away or very close to the camera, (ii) provides extra information about the moving object, making the system capable to discriminate if the moving object might be a human or something else, like a pet, and (iii) improves system's performance for a wider range of camera positions and mountings, since this measure is view-invariant.

3.6 FALL DETECTION ALGORITHM

In order to discriminate fall incidents than other normal everyday activities we adopt a supervised learning approach based on Support Vector Machines (SVM).

The creation of SVM input data is based on the average calculation for each one of the extracted features, see section 3.5, over a time window of length m. Average calculation operates as a smoothing filter that reduces

the effect of noise and errors during features approximation. In our case m is set to 5 frames (200ms). This value, on the one hand, is sufficient for features smoothing, and on the other, allows a successful description of a fall, which lasts 0.9 seconds (Fu et al., 2008), and thus discriminate it than other activities.

Varying time window can increase misclassification error of falls to other humans' activities. Shorter time spans increase false positive rates, while longer time windows increase false negative rates. The former case triggers multiple alarms for the same actual incident, while simultaneously normal activities, which are characterized by sudden changes in features values (e.g. bending), cannot be also considered as falls. The latter case over smooths the values of vertical motion velocity, width-height ratio and body orientation. Thus, it is harder to find an appropriate threshold for discriminating fall incidents than normal activities.

As input data, the average values over 5 frames of (i) the time derivative of the angle that describes body orientation, (ii) the time derivative of person's actual height that represents vertical motion velocity and (iii) the time derivative of person's projected width-height ratio, are used. Each one of the resulted tuples is manually labeled as normal activity or fall.

3.7 EXPERIMENTAL RESULTS

The proposed system works with a single monocular camera. Foreground extraction/background subtraction algorithms and feature extraction processes are implemented in C++ and Python using OpenCV library and the semi-supervised fall detection mechanism is done with Matlab. The system operates in real-time at 23fps for 640×480 frame dimension on an Intel DualCore T4300 at 2.1 GHz by using, for image segmentation, ISL algorithm for indoor environments and ASMM for outdoor. This selection is justified in subsection 7.2. By using these techniques, the fall detection algorithm detects over 90% of fall incidents while it preserves very low false positive rate, which is not crucial when post verification video analysis is available. Its performance and efficiency depends on the quality of extracted features, which inherently depend on the foreground extraction efficiency and persons actual height approximation.

3.7.1 *Data Set Description*

The evaluation of the system performance conducted using footage from a martial arts school in Chania, Greece. The code was implemented in OpenCV. The system was tested in different cases, including camera posi-

Figure 3.5: Characteristics examples of the environment recorded along with the background changes.

Figure 3.6: Examples of different normal humans' activities tested.

tion (meaning that active cameras scenarios can be also supported), changes in the illumination conditions, rapid and fluctuations in the background. We should mention that the results in the martial arts school in Chania were focused on background, where the sun light reflects on mirrors, making the illumination changes really demanding. Furthermore, video background was changing dynamically. New objects were appeared in the scene and existing objects changed position. Below are shown three pictures of background. Figure 3.5 depicts some examples of background changes. In Figure 3.5(a) the curtains are closed while in Figure 3.5(b) the curtains are opened and in Figure 3.5(c) curtains are opened and one bench was appeared in the scene.

Camera calibration approach, described in subsection 3.5.2, is based on the detection of three finite vanishing points that correspond to three mutually orthogonal planes of the scene. Thus, the camera has to be placed in a way that its plane is not parallel to any of the aforementioned planes. Furthermore, in order to calculate 2D and 3D features (subsection 3.5.1 and subsection 3.5.2) for discriminating falls than normal activities, we mounted the camera so that frames' x-axis was almost parallel to the floor. In cases where frames' x-axis is not parallel to the floor, the floor plane has to be detected and an affine transformation must be conducted to reverse rotation effects before the application of the proposed approach. However, computing such an affine transformation is out of the scope of this work.

Experimental process includes several actions such as (a) falls, (b) appearance/disappearance of objects, and (c) normal activities. Falls were made in every direction according to the camera. This includes falls to the right, to the left, with forward and backward motion in regard to the camera position. In total, the dataset contains 50 fall incidents and many more normal activities. Several objects were used, such as benches and balls to simulate normal activities, like sitting or playing with the ball, and falls, like falling from the bench. Normal activities simulated during the experiment, included leaning forward to tie the shoelace, laying down on the floor, sitting on the bench, sitting down on the floor. These normal activities may look like a fall, but

Figure 3.7: (a) Original captured frame, (b) ISL performance, (c) ASMM performance, (d) nPBG performance and (e) GMM performance.

they are not a real fall, so they used to check false negative and positive rates and consequently the performance of the system. Examples of normal activities are shown in Figure 3.6.

3.7.2 *Foreground Extraction*

During experimentation process three different background subtraction techniques were used: ISL algorithm, ASMM, nPBG. All these algorithm were compared to Gaussian Mixture Model (GMM) background subtraction in terms of computational cost, precision and recall. GMM background subtraction technique is also implemented in C++ using OpenCV.

	Overall		**Indoor**		**Outdoor**	
	precision	recall	precision	recall	precision	recall
ASMM	68%	79%	72%	78%	61%	79%
ISL	62%	77%	94%	93%	23%	47%
nPGB	71%	70%	91%	72%	58%	79%
GMM	41%	59%	58%	71%	29%	51%

Table 3.1: Precision and Recall diagrams for indoor-outdoor environments, only indoor environments and only outdoor environments.

Image segmentation techniques performances, for foreground extraction, are summarized in Figure 3.7. Figure 3.7(a) shows the original captured frame and the visual results of ISL, ASMM, nPBG and GMM are presented in Figure 3.7(b-e). The first two columns present segmentation results for indoor environment, columns 3 to 6 present segmentation results for two video sequences from VISOR (http://openvisor.com) and the last two columns present segmentation results for a very challenging video sequence from SCOVIS.

As it is easy to be seen, for indoor environment, the ISL algorithm extracts almost the "perfect" silhouette of foreground object and outperforms all other algorithms, whose detection is characterized by many "holes" on the body of foreground object. However, for outdoor environments ISL algorithm's performance is very weak. This algorithm uses motion information

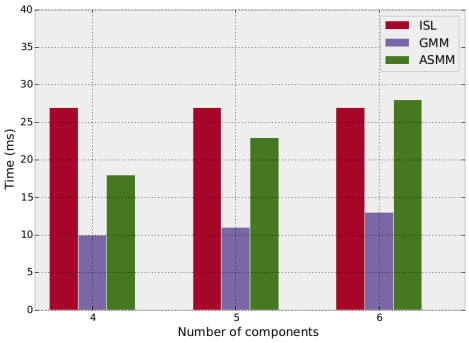


Figure 3.8: Computational cost per frame for different background subtraction methods. Cost for nPBG is not presented as it depends on the area of the foreground object, its average cost is 70ms for 1000 pixels foreground.

in the scene to extract the foreground and its weak performance is justified by the fact that outdoor environments are characterized by high background motion. For outdoor environments visual results suggest that ASMM and nPGB algorithms perform better.

However, visual results are not sufficient to compare algorithms' performance. For this reason, we quantified performances' results in terms of precision and recall and computational cost. For precision and recall computation, background algorithm results were compared to a foreground mask, which was labeled by a human user, was used as the reference silhouette of foreground object.

A precision recall diagram is presented in the left column of Table 3.1, for indoor-outdoor environments performance. ISL, ASMM and nPBG algorithms performs quite well with precision rate over 63% and recall rate over 70%. In order to select the more robust algorithm for our system, we have to examine precision recall diagrams for indoor, Table 3.1 middle column, and outdoor, Table 3.1 right column, environments separately. For indoor environment ISL algorithm outperforms all other algorithms with precision rate 95% and recall rate 94%, while for outdoor environments ASMM presents the best performance with precision rate 61% and recall rate 78%. nPGB algorithm's performance is presented to be independent to operation environment.

Besides algorithms' precision and recall, another requirement is the proposed fall detection system to be capable to operate in real-time. So, an important factor for the selection of appropriate segmentation techniques is their computational cost per frame Figure 3.8. Computational cost of nPBG is not presented in the diagram, as its cost is depended on the size of the area of foreground object. This algorithm presents average cost of 70ms for 1000 pixels area of foreground and thus is not suitable for real-time operation (25fps). ISL algorithm requires 27ms to process each frame and ASMM time requirements vary from 18ms to 29ms depending on the number of Student's-t mixtures. Both these algorithms can operate in real-time.

The aforementioned analysis suggests to use ISL for indoor environments in order to exploit its high precision and recall rates and ASMM for outdoor environments as it outperforms all other algorithms. The following feature extraction process is based on the aforementioned setting (ISL for indoor environments and ASMM for outdoor environments).

3.7.3 *Features for Fall Detection*

The features that are used to discriminate fall incidents than other ordinary activities are vertical motion velocity, based on actual person's height,

person's projected width-height ratio and body orientation. Firstly, we will present person's actual height approximation and then we will show how this features change during a fall incident.

To approximate person's actual height we used Eq(23) with an extra constraint, which reduces wrong estimations when a fall incident occurs and height is significantly changing. According to this constraint, the height $h(i)$ is being updated only if its absolute difference from $h(i-1)$ is smaller than a predefined threshold (in our case 30cm). Diagram in Figure 3.9 (b) shows Root Mean Square Error (RMSE) during person's actual height approximation for different values of κ variable. As this figure shows, the system yields the more robust performance when κ variable is set to 0.8 with RMSE equal to 4.27%. During experiments the person was 193cm tall, so this RMSE corresponds to 8.2cm.

This approximation error caused by the human motion and small errors during foreground extraction and perspective transformations. However, this error is very small and doesn't affect the performance of the system. Approximation of actual person's height is presented in Figure 3.9(a). The horizontal dotted line represents the actual person's height, while the continuous line represents the approximation of its height.

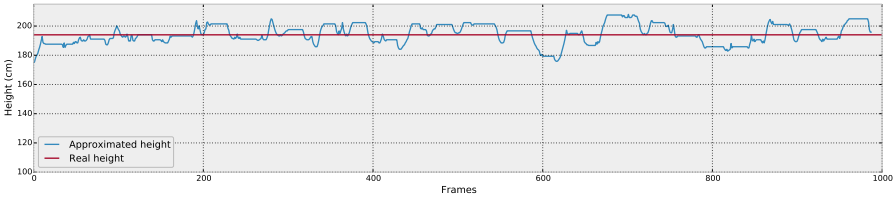
In Figure 3.9(c) time derivatives of features' values are presented. During these frames two fall incidents occurred; one at frame 40 and another one at frame 132. It is easy to be seen that during a fall incident these features present strong time derivatives, which are sufficient to discriminate falls. Instead of following a heuristic or trial-and-error approach for defining thresholds for these derivatives in order to identify falls, we used a supervised learning algorithm. In particular we exploited Support Vector Machines (SVM). Our choice to use SVM is justified by the fact that; a) we want to separate events in two different classes (fall and non-fall class), b) SVM construct a hyperplane that has the largest distance to the nearest training data point of any class decreasing this way the generalization error of the classifier and c) in most of cases, SVM generalization performance (i.e. error rates on test sets) either matches or is significantly better than that of other competing methods.

For the training of the classifier we randomly selected 60% of fall and non fall incidents. Furthermore, we used Gaussian Radial Basis Function kernel, as this configuration leads to the largest functional margin. The training of the classifier takes place offline. For detecting falls every new example is mapped into one of the two classes. If a new example is mapped into the fall class, then a fall alarm occurs.

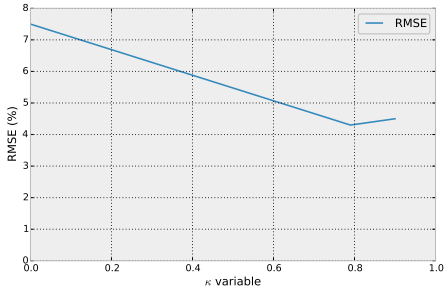
3.7.4 *Fall Detection Algorithm*

During the experimentation process on person simulated falls in every direction according to the camera position, Figure(3.10a), and normal every day activities, that may look like falls but they are not real falls, Figure(3.10b). The fall detection algorithm was tested in dynamically changing visual conditions, including illumination changes, cluttered background and occlusions

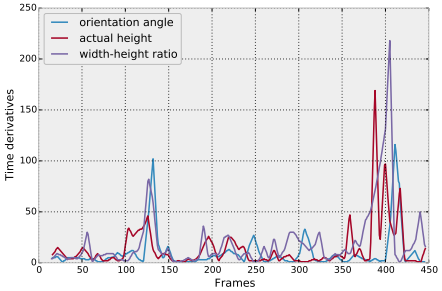
The overall performance of fall detection scheme is presented in Table 3.2. Its performance is affected by the quality of extracted features and subsequently by foreground extraction. For this reason, our system presents more robust performance for indoor environments. However, it should be



(a) Height approximation



(b) RMSE



(c) Features derivatives

Figure 3.9: (a) Actual height approximation for 1000 frames, (b) actual height approximation RMSE in regard to κ variable and (c) features derivatives changes over time.

mentioned that its performance is not affected by humans' height, because the threshold for discriminating fall incidents than normal activities, is estimated through a learning procedure and, thus, is adapted to individual's height. In addition, the impact of occlusions is being reduced as camera's height is being increased.

Finally, in Table 3.3 false positive rates are presented with regard to different activities. The biggest false positive rate is presented when the human lies on the floor, however, this activity cannot be thought as "normal". False positive rates, associated with the "lying on the floor" activity, can decrease by relaxing vertical velocity threshold, used to discriminate fall incidents than normal activities. Relaxing this threshold, however, can increase false negative rates, which are of a primary concern for a fall detection system.

Camera's height (cm)		Proposed system	Indoor No occlusion	Indoor with occlusion	Outdoor
40	*Falls detected*	90 %	98 %	76 %	83 %
	Wrong detections	4	2	5	9
220	*Falls detected*	93 %	97 %	92 %	82 %
	Wrong detections	6	4	7	8
260	*Falls detected*	97 %	97 %	94 %	82 %
	Wrong detections	3	3	4	6

Table 3.2: Proposed System's Overall Performance.

(a) Fall incidents

(b) Normal activities

Figure 3.10: Simulated activities during experimentation process (a) Falls, (b) normal activities.

Activity	False Positive
Lie down	62.5%
Sit on the floor	25%
Other	12.5%

Table 3.3: Total false positive rate divided in regard to human activities

3.8 CONCLUSIONS

This chapter presented a fall detection scheme that uses a single low-cost monocular camera. Our approach combines the advantages of both monocular and multi-camera systems. In contrast to other 2D fall detection methods, the proposed system is very robust for a wider range of camera positions and mountings and its performance is not affected by the distance between camera and foreground objects. Moreover, it extracts 3D features without using computationally expensive stereo-vision mathematics, which are necessary and compulsory for all multiple camera systems.

Specifically, through camera self-calibration and perspective transformations, our system is capable to exploit 3D measures to increase its robustness. It operates in real-time and is capable to detect over 90% of fall incidents in complex and dynamically changing visual conditions, while it presents very low false positive rate. Furthermore, due to its low computational cost and memory requirements making it suitable for large scale implementations, let alone its low financial cost since simple low resolution cameras are used, making it affordable for a large scale.

4

VISION BASED MARITIME SECURITY SYSTEM

4.1 MOTIVATION

Pirate attacks (Onuoha, 2009), unlicensed fishing trailers (Agnew et al., 2009), human trafficking (Stanslas, 2010) and smuggling are only few, among a wide range, of criminal activities known to the maritime domain. Additionally, all ships are vulnerable to problems from weather conditions, faulty design or human errors. Management of such crises and emergency situations can be supported by advanced surveillance systems suitable for complex environments. Indeed, given the enormous size of coast lines and the plethora of vehicle types, effective surveillance is considered extremely hard to obtain, without using the appropriate systems.

The architecture of such systems vary from radar-based to video-based approaches. Radar technology is commonly used in such systems (Zemmari et al., 2013) providing accurate detection results. There are, however, two major drawbacks; it is quite expensive and its performance is highly affected by various factors, such as material of objects and echoes returned from targets, which are out of interest (e.g. the ground, sea, or buildings). Computer vision is an alternative approach, consisting of various techniques, each one with specific advantages and drawbacks. Nevertheless, the majority of such systems are used post-factum and, in most cases, they are controlled by humans, who are responsible for monitoring and evaluating numerous video feeds simultaneously.

The challenge of advanced surveillance systems is to process and present collected sensor data, in an intelligent and meaningful way, to give a sufficient information support to human decision makers (Fischer and Bauer, 2010). Towards this direction, major research effort has been conducted for automatically detecting and tracking vessels at sea through visual cues. Generally, in maritime environments, detection and tracking of vessels is inherently depended on dynamically varying visual conditions (e.g. varying lighting and reflections of sea, wind and rain). So, to successfully design a vision-based surveillance system, we have to carefully define both its operation requirements and vessels' characteristics.

On the one hand, concerning operation requirements, as described in (Szpak and Tapamo, 2011), a system should fulfill specific requirements in order to be of practical use. At first, it must determine possible targets within a scene containing a complex, moving background. Additionally, the system must not produce false negatives and keep as low as possible the number of false positives. Since we are talking about surveillance system, it must be fast and highly efficient, operating at a reasonable frame rate and for long time periods using a minimal number of scene-related assumptions.

On the other hand, the characteristics of vessels at sea vary greatly, making tracking further more difficult. Regardless of variation, there are four major descriptive categories. First comes the size. Vessels size ranges from jet-skis to large cruise ships. Secondly, we have the moving speed. Vessels can be stationary or moving objects with varying speeds. Thirdly, we deal with angle of view. Vessels move to any direction, according to the camera

position, and thus their angle varies from 0°to 360°. Finally, there is vehicles' visibility. Some vessels have a good contrast to the sea water while others are intentionally camouflaged. A robust maritime surveillance system must be able to detect vessels having any of the above properties.

4.2 LITERATURE REVIEW

In this work emphasis is given to visual-based surveillance systems. The main purpose is detection and tracking of targets within camera's range, rather than their trajectory patterns' investigation (Lei, 2013),(Vandecasteele et al., 2013) or their classification in categories of interest (Maresca et al., 2010). The system's main purpose is to support end-user in monitoring coastlines, regardless of existing conditions.

Sanderson in (Sanderson, 1999) proposed a method for object detection, based on anisotropic diffusion, which has high computational cost and performs well only for horizontal and vertical edges. Socek *et al*. (Socek et al., 2005) presented a method that fuses a foreground object detection technique with image color segmentation to improve accuracy. In (Albrecht et al., 2011a),(Albrecht et al., 2010) a maritime surveillance system mainly focuses on finding regions in images, where is a high likelihood of a vessel being present, is proposed. In (Albrecht et al., 2011b), the aforementioned system was expanded by adding a sea/sky classification approach based on Histogram of Oriented Gradients (HOG). The authors of (Rodriguez Sullivan and Shah, 2008) proposed a method for securing port facilities by automatically detecting various vessel classes using a trained set of MACH filters.

All of the above approaches adopt offline learning methods that are sensitive to accumulation errors and difficult to generalize for various vessel types, angles and environmental/visual conditions. Wijnhoven *et al*. (Wijnhoven et al., 2010) utilized an online trained classifier, based on HOG. However, retraining takes place when a human user manually annotates the new training set. In (Szpak and Tapamo, 2011) an adaptive background subtraction technique is proposed for vessels extraction. Unfortunately, when a target is almost homogeneous is difficult, for the background model, to learn such environmental changes without misclassifying the target.

More recent approaches are the works of Makantasis *et al*. (Makantasis et al., 2013) and Kaimakis *et al*. (Kaimakis and Tsapatsoulis, 2013). The former, utilizes a fusion of Visual Attention Maps (VAM) and background subtraction algorithm, based on Mixture Of Gaussians (MOG), to produce a refined VAM. These features are fed to a neural network tracker, which is capable of online adaptation. The latter, utilized statistical modelling of the scene's non-stationary background to detect targets implicitly. Both cases were based on monocular video data and no a priori knowledge about targets' appearance is required

A comparative evaluation of anomaly detection algorithms, for maritime video surveillance can be found in (Auslander et al., 2011). This work emphasizes on algorithms that automatically learn anomaly detection models for maritime vessels, where the tracks are derived from ground-based optical video, and no domain-specific knowledge is employed. Some models for anomaly detection can be created manually, by eliciting anomaly models in the form of rules from experts (Nilsson et al., 2008), but this may be impractical if experts are not available, cannot easily provide these models, or the elicitation cost may be high.

4.2.1 *Our Contribution*

A careful examination of the proposed methodologies on visual maritime surveillance suggests that specific points have to be addressed. Firstly, a system needs to combine both supervised and unsupervised tracking techniques, in order to exploit all the possible advantages. Secondly, since we deal with vast amount of available data, we need to reduce, as much as possible, the required effort for the initialization of the system. Finally, there is the adaptation problem on constantly varying visual environments.

The innovation of this work lies in the creation of a visual detection system, able to overcome the aforementioned difficulties by combining various, well tested techniques and, at the same time, minimizes effort during the offline initialization using a Semi-Supervised Learning (SSL) approach, appropriate for large data sets. Thus, the first step, towards the creation of a robust detection system, would be the utilization of unsupervised techniques. Collaboration of visual attention maps, that represents the probability of a vessel being present in the scene, and background subtraction algorithms further supports image segmentation and excludes any land parts marked as vessels. Regarding the supervised technique Support Vector Machines (SVMs) have been used.

Given a set of frames, the manual segmentation requires a lot of effort and time. In order to facilitate the creation of such training set, SSL graph-based algorithms need to be involved. Unfortunately, SSL techniques scale badly as the available data rises. To make matters worse, Nadler *et al.* (Nadler et al., 2009) have shown that graph Laplacian methods, and more specific the regularization approach (Zhu et al., 2003) and the spectral approach (Belkin and Niyogi, 2003), are not well posed in spaces $\mathbb{R}^d$, $d \geq 2$, and as the number of unlabeled points increases the solution degenerates to a non-informative function. Consequently, a semi-supervised procedure, suitable for large data sets is exploited for the offline initialization, significantly reducing the effort required.

The proposed system is able to operate in real-time for long time periods such as months without any re-initialization. In addition, camera motion does not affect system's false negative rate, which is the most important characteristic in this application domain. Also, the system does not make any assumptions related to scene, environment and/or visual conditions. An important aspect is the minimum labeling effort, for the training set creation, required by the trackers since the semi-supervised technique nullifies segmentation errors.

4.3 SYSTEM ARCHITECTURE AND PROBLEM FORMULATION

In this section we analytically describe the architecture of the maritime security system and formulate the problem of detecting and tracking maritime targets.

4.3.1 *System Architecture*

The goal of the presented system is the real-time detection and tracking of maritime targets. Towards this direction, an *appearance-based* approach is adopted to create visual attention maps that represent the probability of a

target being present in the scene. High probability implies high confidence for a maritime target's presence.

Visual attention maps creation is based exclusively on each frame's visual content. Consequently, they do not take into consideration neither the temporal relationship between subsequent frames, nor any motion information presented in the scene. Due to this limitation, in many cases, high probability is assigned to image regions that depict, instead of maritime targets, stationary land parts, which stand out in relation to their surrounding regions or the entire image, such as port facilities, lighthouses, rocks, etc.

In order to overcome such a drawback, our system exploits the temporal relationship between subsequent frames. Concretely, video blocks, containing a predefined number of frames, are used to model the pixels' intensities. Thus, the temporal evolution of pixels intensities is utilized to estimate a pixel-wise background model, capable to denote each one of the pixels of the scene as background or foreground. By using a background modeling algorithm, the proposed system can efficiently discriminate moving from stationary objects in the scene. In order to model pixels' intensities, we use the background modeling algorithm presented in (Zivkovic, 2004). This choice is justified by the fact that this algorithm can automatically fully adapt to dynamically changing visual conditions and cluttered background.

Let us denote as $p_{xy}^{(i)}$ the pixel of a frame i at location (x, y) on image plane. Having constructed the visual attention maps and applied background modeling algorithm, the pixel $p_{xy}^{(i)}$ is described by a feature vector

$$\boldsymbol{f}_{xy}^{(i)} = [f_{1,xy}^{(i)} \cdots f_{k,xy}^{(i)}]^T \ , \tag{4.1}$$

where $f_{1,xy}^{(i)}, \cdots, f_{k-1,xy}^{(i)}$ stand for scalar features that correspond to the probabilities assigned to the pixel $p_{xy}^{(i)}$ by different visual attention maps, while $f_{k,xy}^{(i)}$ is the binary output of background modeling algorithm, associated with the same pixel. In order to detect maritime targets, these features are fed to a binary classifier which classifies pixels into two disjoint classes, C_T and C_B.

If we denote as $Z^{(i)} = C_T^{(i)} \cup C_B^{(i)}$ the set that contains all pixels of frame i, then the first class, $C_T^{(i)}$, contains all pixels that depict a part of a maritime target, while the second class, $C_B^{(i)}$, equals to $Z^{(i)} - C_T^{(i)}$. We used SVMs to transact the classification task for the proposed maritime surveillance system. Selection of the SVM, over other supervised classification methods, is justified by its robustness, when handling unbalanced classes.

The overall architecture of the proposed maritime surveillance system is presented in Figure(4.1). Initially, the original captured frame, Figure(4.1-i), is processed to extract pixel-wise features using visual attention maps, Figure(4.1-ii), and background modeling, Figure(4.1-iii). Then the feature vector of each one pixel, Figure(4.1-iv), is processed by a binary classifier, Figure(4.1-v), who decides if the pixel corresponds to a part of a maritime target, Figure(4.1-vi). The classifier's output in frame-level is shown in Figure(4.1-vii).

4.3.2 *Problem Formulation*

Maritime target detection can be seen as an image classification problem. Thus, we classify each one of the frame's pixels in one of two classes, C_T

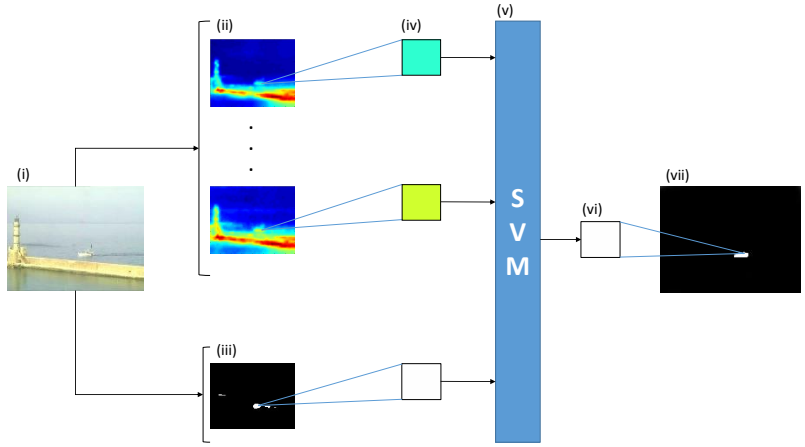


Figure 4.1: System's architecture illustration. Image in (i), corresponds to the original captured frame. In (ii), the output of visual attention maps is presented. High probability is represented with red color, while low probability with deep blue. The output of background modeling algorithm is shown in (iii). The column in (iv) represents a feature vector for a specific pixel, which is fed to a binary classifier (v). The output of the classifier in pixel level is presented in (vi) and in frame level in (vii).

and C_B. If we denote as $l_{xy}^{(i)}$ the label of pixel $p_{xy}^{(i)}$, then, for a frame i, the classification task can be formulated as

$$l_{xy}^{(i)} = \begin{cases} 1 & \text{if } p_{xy}^{(i)} \in C_T \text{ for } x = 1,...,w \text{ and } y = 1,...,h \\ -1 & \text{if } p_{xy}^{(i)} \in C_B \text{ for } x = 1,...,w \text{ and } y = 1,...,h \end{cases}, \quad (4.2)$$

where h and w stand for frame's height and width.

Although, a binary classifier, in our case SVM, can successfully transact the classification task, a classifier training process should precede. Training process requires the formation of a robust training set composed of pixels, along with their associated labels. Such a set can be formed by the user, through a rough segmentation of a frame t into two regions, that contain positive and negative samples, i.e. pixels that belong to $C_T^{(t)}$ class and labelled with 1 and pixels that belong to $C_B^{(t)}$ class and labelled with -1. The union of $C_T^{(t)}$ and $C_B^{(t)}$ consists the initial training set S.

At this point in the training set $S = \{(p_{xy}^{(t)}, l_{xy}^{(t)})\}$ for $x = 1,...,w$ and $y = 1,...,h$, each pixel is described only by its intensity, which does not provide sufficient information for separating pixels into two disjoint classes. Taking into consideration the application domain, which indicates that the largest part of a frame will depict sea and sky, we exploit low level features to emphasize man-made structures in the scene.

Based on image low level features, we create visual attention maps that indicate the probability a pixel to depict a part of a maritime target. In addition, based on the observation that a vessel must be depicted as a moving object, we implicitly capture the presence of motion by exploiting a background modeling algorithm. Using the output of visual attention maps and

the background modeling algorithm, each pixel is described by the feature vector of Eq.(4.1) and the training set S can be transformed to

$$S = \{(\boldsymbol{f}_{xy}^{(t)}, l_{xy}^{(t)})\} \quad \text{for} \quad x = 1, ..., w \text{ and } y = 1, ..., h \ . \tag{4.3}$$

Although the elements of S are labelled by a human user, the labelling procedure may contain inconsistencies. This is mainly caused by the fact that human centric labelling, especially of image data, is an arduous and inconsistent task, due to the complexity of the visual content and the huge manual effort required.

To overcome this drawback, we refine the initial training set by i) selecting the most *representative* samples from each class and ii) labelling the rest of the samples using a *semi-supervised* algorithm. Selection of the most representative samples is taken place by applying simplex volume expansion on the samples of each class separately. Then representative samples are used by the semi-supervised algorithm as landmarks, in order to label the rest of the samples. Using the refined training set, the binary classifier can be successfully trained to classify the pixels of subsequent frames, addressing this way the initial classification problem of Eq.(4.2).

4.4 PIXEL-WISE VISUAL DESCRIPTION

In this section we describe the procedure for constructing feature vectors, capable to characterize the pixels of a frame. During feature vectors construction emphasis is put on the application domain and the specific structure and appearance of maritime objects. The whole process is tuned for maritime imagery and is guided by the operational requirements that an accurate and robust maritime surveillance system must fulfill. Feature vectors are created for each pixel.

4.4.1 *Scale Invariance*

Potential targets in maritime environment vary in sizes, either due to their physical size or due to the distance between them and the camera. Despite that, most of the feature detectors operate as kernel based method and thus they prefer objects of a certain size. As presented in (Alexe et al., 2010) and (Liu et al., 2011) images must be represented in different scales in order to overcome this limitation. In our approach, a Gaussian image pyramid is exploited in order to provide scale invariance and to take into consideration the relationship between adjacent pixels.

The Gaussian image pyramid is created by successively low-pass filtering and sub-sampling an image. During the stage of low-pass filtering the Gaussian function can be approximated by a discretized convolution kernel

$$\boldsymbol{G}_d = \frac{1}{256} \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix} \ . \tag{4.4}$$

The idea of creating a Gaussian kernel for convolution is to use a 2D Normal distribution as a point-spread function. Since the image is stored as

a collection of discrete pixels we need to produce a discrete approximation of the Gaussian function. In theory, the Gaussian distribution is non-zero everywhere, which would require an infinitely large convolution kernel, but in practice it is effectively zero more than about three standard deviations from the mean, and so we can truncate the kernel at this point. However, it is not obvious how to pick the values of the kernel to approximate a Gaussian. One could use the value of the Gaussian at the center of a pixel in the mask, but this is not accurate because the value of the Gaussian varies non-linearly across the pixel. For this reason, the value of the Gaussian is integrated over the whole pixel. Because the integrals are not integers the array is rescaled so that the corners had the value 1. Finally, 256 is the sum of all values in the kernel.

During sub-sampling every even-numbered row and column is removed. If we denote as I^o the original captured image and as I^ϕ the image at pyramid level ϕ then image at pyramid level $\phi + 1$ is computed as

$$I^{\phi+1}(x, y) = [\boldsymbol{G}_d * I^\phi](2x, 2y) \ . \tag{4.5}$$

One must combine the various scales together into a single unified and scale-independent feature map, to provide scale-independent feature analysis. To do so, image at level $\phi + 1$, firstly, is upsized twice in each dimension, with the new even rows and columns filled with zeros. Secondly, a convolution is performed with the kernel $\boldsymbol{G}_u$ to approximate the values of the "missing pixels". Because each new pixel has four non new-created adjacent pixels, $\boldsymbol{G}_u$ is defined as

$$\boldsymbol{G}_u = 4 \cdot \boldsymbol{G}_d \ . \tag{4.6}$$

Then, a pixel-wise weighted summation is performed to adjacent images in pyramid so as the unified image J^ϕ at level ϕ is defined as

$$J^\phi = \frac{1}{2} \cdot [I^\phi + [\boldsymbol{G}_u * U(I^{\phi+1})]] \ , \tag{4.7}$$

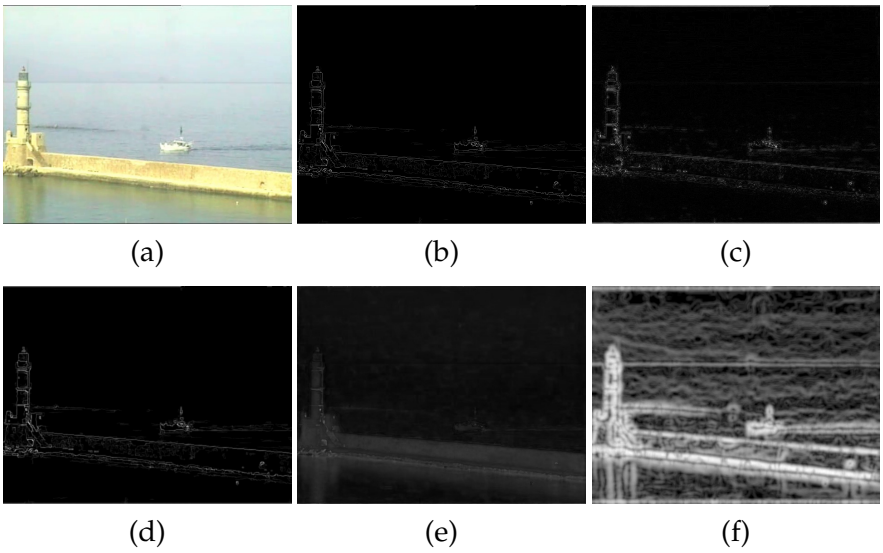
where U stands for the upsize operation. The final unified image is computed by repeating the above operation, from coarser to finer pyramid image levels.

4.4.2 *Low-level Features Analysis*

As described in (Albrecht et al., 2010), (Albrecht et al., 2011a) and (Albrecht et al., 2011b) different low-level image features respond to different attributes of potential maritime targets. Thus, a combination of features should be exploited in order to reveal targets' presence. These are edges, horizontal and vertical lines, frequency, color and entropy.

The density of image edges can successfully describe the overall structure of an image, horizontal and vertical lines are able to denote man-made structures, making the system able to suppress large image regions, depicting sea and sky. Frequency can successfully emphasize objects in noisy conditions, such as vessels in a wavy sea. Color feature can successfully emphasize objects colored different than sea and sky and, finally, entropy quantifies the amount of information coded in an image.

Each one of these features are calculated for all image's pyramid levels, independently. Then, image's pyramid is combined to form a single unified



(a) (b) (c)

(d) (e) (f)

Figure 4.2: Original captured frame (a) and feature responses (b)-(f); (b) edges, (c) frequencies, (d) vertical and horizontal lines, (e) color and (f) entropy. All feature responded to the land part and the boat (maritime target).

feature map by using Eq.(4.7). In Figure(4.2) the original captured frame along with the features responses are presented. All of the features emphasize the stationary land part and the white boat, which is the actual maritime target.

4.4.2.1 *Edges*

In order to successfully exploit images' edges, the system must be able to detect them in a very accurate way and, at the same time, preserve their magnitude. For this reason edge detection is taking place by combining both Canny and Sobel operators.

Canny operator (McIlhagga, 2010) is a very accurate image edge detector, which outputs zeros and ones for image edges absence and presence respectively. Sobel operator (Yasri et al., 2008), although being less accurate, measures the strength of detected edges by approximating the derivatives of intensity changes along image rows and columns.

So, by multiplying pixel-wise the output of two operators the system is able to detect edges in a very accurate way, while at the same time it preserves their magnitude. If we denote as C_I and S_I the Canny and Sobel operators for image I, then the edges $\boldsymbol{\mathcal{E}}_I$ are defined as

$$\boldsymbol{\mathcal{E}}_I = C_I \cdot S_I \ . \tag{4.8}$$

Matrix $\boldsymbol{\mathcal{E}}_I$ has the same dimensions with image I; its elements $\boldsymbol{\mathcal{E}}_I(x, y)$ correspond to the magnitude of an image edge at location (x, y) on image plane.

4.4.2.2 *Frequency*

Frequency feature is utilized to emphasize regions of high frequency. For computing the high frequency components of the input frame, I, the following relation is used,

$$\boldsymbol{\mathcal{F}}_I = \nabla^2 \cdot I \ . \tag{4.9}$$

The matrix $\boldsymbol{\mathcal{F}}_I$ has the same dimensions with image I and its elements $\boldsymbol{\mathcal{F}}_I(x, y)$ correspond to the frequency's magnitude at location (x, y) on image plane.

While exploitation of frequency features may emphasize (highly) wavy sea regions, they will suppress image regions that depict sky parts, since such image parts are dominated by low frequencies. Furthermore, image frequencies are complementary to image edges emphasizing highly structured regions within an object and thus improving detection accuracy.

4.4.2.3 *Horizontal and vertical lines*

The detection of horizontal and vertical lines in an image require an appropriate kernel $\boldsymbol{K}$. Kernel $\boldsymbol{K}$ is tuned to strengthen the response of a pixel if this consists a part of a horizontal or vertical line and suppress pixels' responses in all other cases. In order to emphasize this kind of lines the kernel $\boldsymbol{K}$ is designed as

$$\boldsymbol{K} = \frac{1}{16} \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 1 & 2 & 4 & 2 & 1 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \tag{4.10}$$

and vertical and horizontal lines in a frame I can be computed by

$$\boldsymbol{\mathcal{L}}_I = \boldsymbol{K} * I \ . \tag{4.11}$$

Again, the matrix $\boldsymbol{\mathcal{L}}_I$ has the same dimensions with image I and its elements $\boldsymbol{\mathcal{L}}_I(x, y)$ indicates the magnitude of an horizontal and/or vertical line at location (x, y) on image plane.

Line detector works like an edge detector. In coastal regions, captured frames are likely to contains land parts that will respond to the edge detector, affecting detection accuracy of actual maritime targets. Taking into consideration that vertical and horizontal lines are more dominant, in man-made structures than in natural scenes, the aforementioned line detector can improve detection accuracy of actual targets by suppressing the regions of the image that depict natural land parts, such as rocks.

4.4.2.4 *Color*

In maritime environment, no assumptions about the color of vessels can be made. For this reason instead of focusing on a specific color, differences in color are more likely to indicate the presence of a potential target. Furthermore, maritime scenes usually contain large regions with similar colors (sea and sky). This observation as described in (Achanta et al., 2009) and

(Achanta and Susstrunk, 2010) can be exploited to increase the performance of visual attention maps by identifying potential targets.

In order to compute the color difference, the captured frame's colorspace is converted to CIELab, in which the perceptive color difference between two different colors corresponds to the Euclidean distance between the vectors (L, a and b channels) that represent these colors. The computation of color differences takes place by calculating the Euclidean distances between individual pixels color vectors and the mean color vector of the whole frame. For a frame I this procedure results in a matrix $\boldsymbol{\mathcal{C}}_I$ of the same dimensions, whose element, $\boldsymbol{\mathcal{C}}_I(x,y)$, at location (x,y) on image plane, indicates the difference in color between this pixel and the mean color of the rest pixels of the frame.

4.4.2.5 *Entropy*

The entropy quantifies the information coded in an image. Images that depict large homogeneous regions, such as sky or sea regions, present low entropy, while highly textured images will present high entropy. Image entropy can be interpreted as a statistical measure of randomness, which can be used to characterize the texture of the input image. Thus, entropy can be utilized to suppress homogeneous regions of sea and sky and highlight potential maritime targets. Entropy of a region r of an image is defined as

$$H_r = \sum_{j=1}^{k} P_j^{(r)} \cdot \log P_j^{(r)} \ , \tag{4.12}$$

where $P_j^{(r)}$ is the frequency of intensity j in image region r. For a grayscale image, variable k is equal to 256.

In order to compute entropy for a pixel located at (x,y) on image plane, we apply the relation of Eq.(4.12) on a square window centered at (x,y). In our case, the size of the window is 5×5 pixels. The application of Eq.(4.12) on (x,y) of a frame I, for $x = 1,...,w$ and $y = 1,...,h$, where w and h correspond to frame's width and height, results in a matrix $\boldsymbol{\mathcal{H}}_I$ that has the same dimensions with the frame I. The matrix $\boldsymbol{\mathcal{H}}_I$ can be interpreted as a pixel-wise entropy indicator for frame I.

4.4.3 *Visual Descriptors*

Visual descriptors are computed to encode visual information of captured images, using the extracted low-level features described in Subsection 4.4.2. These descriptors are utilized for constructing the visual attention maps. Their computation, instead of pixel-wise, takes place block-wise, in order to reduce the effect of noisy pixels during low-level features extraction. In this work, like (Albrecht et al., 2010), (Albrecht et al., 2011a) and (Albrecht et al., 2011b), three different descriptors are computed:

a) *Local descriptors* that take into consideration each one of the image blocks separately. Local descriptors indicate the magnitude of local features for each one of image blocks.

b) *Global descriptors* that are capable to emphasize blocks with high uniqueness compared to the rest of the image. To achieve this they indicate how different local features for a specific block are, in relation with the same features of all other image blocks.

c) *Window descriptors* that compare local features of a block with the same features of its neighboring blocks.

4.4.3.1 *Local descriptor*

One local descriptor is computed for each one of the extracted low-level features. Let us denote as F the feature in question, which can correspond to image edges, frequency, horizontal and vertical lines, color or entropy. For the feature F, the computation of local descriptor is derived by feature's response image. As mentioned before, descriptors are computed block-wise. So, firstly the feature's response image is divided into B blocks of size 8×8 pixels. Then, the local descriptor for a specific block j is defined as the average magnitude of the feature F in the same block. More formally, for a block j, with b_h height and b_w width, the local descriptor of feature F is computed by

$$lF_j = \frac{1}{b_h \cdot b_w} \sum_{(x,y)\in j} F(x,y) \ , \tag{4.13}$$

where $F(x,y)$ is the response of feature in question at pixel (x,y). This kind of descriptor is capable to highlight image blocks with high feature responses.

4.4.3.2 *Global descriptor*

The local descriptors handle each image block separately and, thus, are insufficient to provide useful information when features' responses are quite similar along all image blocks. Consider, for example, an image full of edges, like a wavy sea. In this case, local descriptor $l\mathcal{E}$, which is associated with the image edges, is not able to provide useful information about salient objects in the scene, since all blocks will present high edge responses. The proposed system can overcome this problem by using global descriptors.

Uniqueness of a block j can be evaluated by the absolute difference of the feature response between this block and the rest blocks of the image. The global descriptor for a feature F and image block j is calculated by

$$gF_j = \frac{1}{B} \cdot \sum_{i=1}^{B} |lF_j - lF_i| \ . \tag{4.14}$$

As mentioned before a global descriptor is able to emphasize blocks presenting high uniqueness, in term of features' responses, compared to the rest blocks of the image.

4.4.3.3 *Window descriptor*

Local and global descriptors are capable to emphasize image blocks that are highly distinctive, in terms of features' responses, or have a unique presence in the image. However, if potential targets are presented in more than one blocks the aforementioned descriptors will emphasize the most dominant target and will suppress the others. In order to overcome this problem, our system exploits a window descriptor, that compares each image block with its neighboring blocks.

Window descriptor for an image with $N \times M$ blocks uses an image window W, which is spanned by the maximum symmetric distance, d_h and

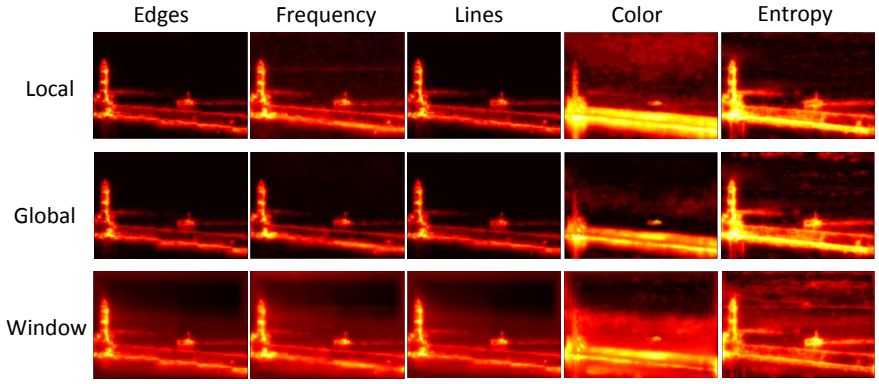


Figure 4.3: Visual attention maps for local, global and window descriptors. Using five low level features and three descriptor, each one of the frame's pixels is described by a 15-dimensional vector. The presented visual attention maps correspond to the original frame of Figure(4.2).

d_v along horizontal and vertical axes respectively. Symmetric distances are defined as $d_h = min(l, k_h, N - k_h)$ and $d_v = min(l, k_v, M - k_v)$, where l is the default symmetric distance, 3 blocks in our case, and k_h and k_v stands for block coordinates on image plane along horizontal and vertical axes respectively. The window descriptor for a feature F and image block j with coordinates (j_1, j_2) is computed by

$$wF_j = \frac{1}{2d_h \cdot 2d_v} \cdot \sum_{k=-d_h}^{d_h} \sum_{l=-d_v}^{d_v} |lF_j - lF_{j_1+k, j_2+l}| \ . \tag{4.15}$$

By using three descriptors and five low-level image features, each image block is described by a 1×15 feature vector. Each feature of this vector corresponds to a different visual attention map. For blocks of size 8×8 pixels the visual attention maps are sixty four times smaller than the original captured frame. In order to create a pixel-wise feature vector, as defined in Eq.(4.1), visual attention maps must have the same dimensions with the original captured frame. For this reason they are upsampled, by using Eq.(4.7).

Visual attention maps that correspond to the original frame of Figure(4.2), for each one of the descriptors and each one of the low level features, are presented in Figure(4.3). All visual attention maps emphasize the stationary land part and the boat, while at the same time they suppress the background.

4.4.4 *Background Subtraction*

For the maritime surveillance case, most state-of-the-art background modeling algorithms, like (Doulamis and Doulamis, 2012), fail either due to their high computational cost or due to the continuously moving background, and moving cameras. However, if the background modeling algorithm output is fused in a unified feature vector with the previously constructed visual attention maps, our system will be able to emphasize potential threats and at the same time to suppress land parts that may be appeared in the scene by implicitly capture motion presence.

The proposed system uses the Mixtures of Gaussians (MOG) background modeling technique, presented in (Zivkovic, 2004). This choice is justified by the fact that MOG is fast, robust to small periodic movements of background, and easy to parameterize algorithm. By fusing together the outputs of visual attention maps and the output of a background modeling algorithm, camera motion temporarily increases false positives detections, but false negatives, that comprises the most important characteristic of a maritime surveillance system, are not affected.

For the sake of completeness, we briefly describe the background modeling technique we are using. The goal of any background modeling technique is to decide if a pixel at time t, $x^{(t)}$, belongs to background (BG) or to foreground (FG). A reliable decision can be made if the probability of pixel $x^{(t)}$ to belong to BG is bigger than a threshold. This can be expressed as

$$p(x^{(t)}|BG) > c_{thr} \ . \tag{4.16}$$

The probability $p(x^{(t)}|BG)$ consists the background model, which can be estimated via a training set $X_T = \{x^{(t)}, x^{(t-1)}, ..., x^{(t-T)}\}$ defined over a time span T. The estimated background model is denoted as $\hat{p}(x^{(t)}|X_t, BG)$. For each new pixel sample the training set X_T is updated and $\hat{p}(x^{(t)}|X_t, BG)$ is re-estimated. Fitting the data of X_T using a mixture model of M Gaussian components we have

$$\hat{p}(x^{(t)}|X_T, BG + FG) = \sum_{m=1}^{M} \hat{\pi}_m \mathcal{N}(x^{(t)}; \hat{\mu}_m, \hat{\sigma}_m^2 I) \ , \tag{4.17}$$

where $\hat{\mu}_1, ..., \hat{\mu}_M$ are the estimates of the means and $\hat{\sigma}_1, ..., \hat{\sigma}_M$ are the estimates of the variances that describe the Gaussian components. The non-negative mixing weights are denoted as $\hat{\pi}_m$ and sum up to one. In conditional probability, $BG + GF$ is used because among the samples of X_T could be some values that belong to the foreground objects. The samples of X_T, which correspond to background, consist the majority in the set and their values change gradually. Gaussian components associated with background samples will present larger mixing weight $\hat{\pi}_m$ compared to the samples that are associated with foreground samples. Therefore, the background model of Eq.(4.16) can be approximated as

$$\hat{p}(x^{(t)}|X_T, BG) \sim \sum_{m=1}^{B} \hat{\pi}_m \mathcal{N}(x^{(t)}; \hat{\mu}_m, \hat{\sigma}_m^2 I) \ . \tag{4.18}$$

If the components are sorted to have descending weights $\hat{\pi}_m$, then B is estimated as

$$B = \arg\min_b \left(\sum_{m=1}^{b} \hat{\pi}_m > (1 - c_f) \right) \ , \tag{4.19}$$

where c_f is a measure of the maximum portion of the data that can belong to foreground objects without influencing the background model. The output of background modeling algorithm is presented in Figure(4.4).

4.5 TARGET DETECTION VIA PIXEL-WISE BINARY CLASSIFICATION

The maritime target detection can be seen as an image segmentation problem. In our case target detection, is further reduced to a binary classification

(a) (b)

Figure 4.4: Original frame (a) and the output of background modeling algorithm (b).

problem. For any pixel at location (x, y) of a frame i, the feature extraction process (see Section 4.4) constructs an 1×16 feature vector, $\boldsymbol{f}_{xy}^{(i)}$. Given $\boldsymbol{f}_{xy}^{(i)}$ as input, the classifier will decide if the corresponding pixel depicts some part of a maritime target or not.

4.5.1 *Initial Training Set Formation*

In order to be able to exploit a binary classifier, a process of classifier training should be preceded. Training process requires the formation of a robust training set which contains pixels along with their associated labels. Let us denote as $Z^{(t)}$ the set that contains all the pixels of frame t, $C_T^{(t)}$ the set that contains pixels that depict some part of a maritime target and as $C_B^{(t)}$ the set $Z^{(t)} - C_T^{(t)}$.

The creation of a training set S requires from the user to roughly segment the frame t into two regions, which contain positive and negative samples (i.e. pixels that belong to $C_T^{(t)}$ and $C_B^{(t)}$ class respectively). This segmentation results in a set $S = \{(p_{xy}^{(t)}, l_{xy}^{(t)})\}$, and labels

$$l_{xy} = \begin{cases} 1 & \text{if} \quad p_{xy} \in C_T \\ -1 & \text{if} \quad p_{xy} \in C_B \end{cases} , \tag{4.20}$$

where p_{xy} is a pixel at location (x, y). By utilizing the feature vector $\boldsymbol{f}_{xy}^{(t)}$ the set S takes the form of Eq.(4.3).

However, human centric labeling, especially of image data, is an arduous and inconsistent task, mainly due to the complexity of the visual content and the huge manual effort required. To overcome this drawback, we refine the initial training set through a *semi-supervised* approach.

4.5.2 *Semi-supervised Training Set Refinement*

In order to refine the initial user-defined training set, we partition the set S into two disjoint classes, R and U. The class R contains the most representative samples of S, i.e. the samples that can best describe the classes C_T and C_B, while class U is equal to $S - R$. Samples of class R are considered as la-

beled, while samples belonging to U are considered as unlabeled. Then, via a semi-supervised approach the samples of R are used for label propagation through the ambiguously labeled data of U. In the following we describe in detail the aforementioned process.

4.5.3 *Representatives Selection through Simplex Volume Expansion*

For selecting the most representative samples for each one of the classes C_T and C_B, we consider each sample as a point into an μ-dimensional space. In our case μ is equal to 16, because the dimension of the space is equal to the dimension of the feature vectors that describe the pixels. The process for representatives selection is conducted twice, once for class C_T and once for C_B. In the following we describe the process for representative samples selection for one of the classes, let's say C_T. Exactly the same process is followed for selecting representatives for the other class.

We assume that the μ-dimensional volume formed by a simplex with vertices specified by the most representative points (pixels), belonging to class C_T, should be larger than that formed by any other combination of points of the same class. Let us denote as $\boldsymbol{\nu}^{(i)}$ the i^{th} representative sample, as β the number of representatives to be generated, as $C_{T,R} = \{\boldsymbol{\nu}^{(1)}, \boldsymbol{\nu}^{(2)}, ..., \boldsymbol{\nu}^{(\beta)}\} \subseteq C_T$ the set that contains the representative samples and as $\boldsymbol{w}^{(j)}$ the vector that equals to $\boldsymbol{\nu}^{(j)} - \boldsymbol{\nu}^{(1)}$ for $j = 2, 3, ..., \beta$. Then the volume, $V(C_{T,R})$, of the simplex whose vertices are the points $\boldsymbol{\nu}^{(i)}$ for $i = 1, 2, ..., \beta$ can be computed by

$$V(C_{T,R}) = \frac{|\det(\boldsymbol{W}\boldsymbol{W}^T)|^{1/2}}{(\beta-1)!} \ , \tag{4.21}$$

where $\boldsymbol{W}$ is an $(\beta-1) \times \mu$ matrix whose rows are the row vectors $\boldsymbol{w}^{(j)}$.

The estimation process involves several steps. Initially the set $C_{T,R}$ is constructed by randomly selecting β samples from set C_T and the volume of the simplex, formed by the elements of $C_{T,R}$, is calculated. Then, an iterative approach is adopted to test every sample in the set C_T as a candidate representative. Each one of the samples of $C_{T,R}$ is replaced, one at a time, with a sample $\hat{\boldsymbol{\nu}}$ from C_T that is being tested as candidate representative. Then, the algorithm evaluates if replacing any of the elements, of $C_{T,R}$ with the sample being tested, results in a larger simplex volume. If this is true, let's say for the point $\boldsymbol{\nu}^{(j)} \in C_{T,R}$, then the $\boldsymbol{\nu}^{(j)}$ point is replaced by the image point $\hat{\boldsymbol{\nu}}$ and the process is repeated again until each one of the samples of C_T set is evaluated.

The selection method is *scalable* to large datasets, using an incremental approach. Let us assume that β representatives are known. Then, the problem of selecting $\beta+1$ representatives can be reduced to finding $\beta+1$ representatives *given* β of them. This way, only the volumes of simplices formed by the elements of the sets $C_{T,R} \cup \boldsymbol{x}^{(i)}$ for $\boldsymbol{x}^{(i)} \in C_T$ need to be evaluated.

4.5.4 *Graph-based Semi-supervised Label Propagation*

The aforementioned procedure results to two sets of representative samples, $C_{T,R}$ and $C_{B,R}$, one for each class. The samples of $C_{T,R}$ and $C_{B,R}$ are

considered as labeled, while the rest samples of the classes C_T and C_B are considered as ambiguously labeled. More formally, we have

$$R = C_{T,R} \cup C_{B,R} \tag{4.22}$$

and

$$U = S - C_{T,R} - C_{B,R} \ . \tag{4.23}$$

At this point, we need to refine the initial training set, S, using a suitable approach for the label propagation, through the ambiguously labeled data.

Thus, we need to estimate a labeling prediction function $g : \mathbb{R}^{\mu} \mapsto \{-1, 1\}$ defined on the samples of S, by using the labeled data R. Let us denote as $\boldsymbol{r}_i$ the samples of set R such as $R = \{\boldsymbol{r}_i\}_{i=1}^{m}$, where m is the cardinality of the set R. Then, according to (Liu et al.), the label prediction function can be expressed as a convex combination of the labels of a subset of representative samples

$$g(\boldsymbol{f}_i) = \sum_{k=1}^{m} Z_{ik} \cdot g(\boldsymbol{l}_k) \ , \tag{4.24}$$

where Z_{ik} denotes sample-adaptive weights, which must satisfy the constraints $\sum_{k=1}^{m} Z_{ik} = 1$ and $Z_{ik} \geq 0$ (convex combination constraints). By defining vectors $\boldsymbol{g}$ and $\boldsymbol{\alpha}$ respectively as $\boldsymbol{g} = [g(\boldsymbol{f}_1), ..., g(\boldsymbol{f}_n)]^T$ and $\boldsymbol{\alpha} = [g(\boldsymbol{r}_1), ..., g(\boldsymbol{r}_m)]^T$ Eq.(4.24) can be rewritten as $\boldsymbol{g} = \mathbf{Z}\boldsymbol{\alpha}$ where $\mathbf{Z} \in \mathbb{R}^{n \times m}$, where n is the number of samples belonging to S.

The designing of matrix $\mathbf{Z}$, which measures the underlying relationship between the samples of U and representative samples R (were $R \subset U$), is based on weights optimization; actually non-parametric regression is being performed by means of data reconstruction with representative samples. Thus, the reconstruction for any data point $\boldsymbol{f}_i, i = 1, ..., n$ is a convex combination of its closest representative samples. In order to optimize these coefficients the following quadratic programming problem needs to be solved.

$$\begin{aligned} \min_{\boldsymbol{z}_i \in \mathbb{R}^s} \quad & h(\boldsymbol{z}_i) = \frac{1}{2} ||\boldsymbol{f}_i - \mathbf{R}_s \cdot \boldsymbol{z}_i||^2 \\ \text{s.t.} \quad & \mathbf{1}^T \boldsymbol{z}_i = 1, \boldsymbol{z}_i \geq 0 \ , \end{aligned} \tag{4.25}$$

where, $\mathbf{R}_s \in \mathbb{R}^{\mu \times s}$ is a matrix containing as elements a subset of $R = \{\boldsymbol{r}_1, ..., \boldsymbol{r}_m\}$ composed of $s < m$ nearest representative samples of $\boldsymbol{f}_i$ and $\boldsymbol{z}_i$ stands for the i^{th} row of $\mathbf{Z}$ matrix.

Nevertheless, the creation of matrix $\mathbf{Z}$ is not sufficient for labeling the entire data set, as it does not assure a smooth function g. As mentioned before, a large portion of data are considered as ambiguously labeled. Despite the small labeled set, there is always the possibility of inconsistencies in segmentation; in specific frames the user may miss some pixels that depict targets. In order to deal with such cases the following SSL framework is employed:

$$\min_{\boldsymbol{A} = [\boldsymbol{\alpha}_1, ..., \boldsymbol{\alpha}_c]} \mathcal{Q}(\mathbf{A}) = \frac{1}{2} ||\mathbf{Z} \cdot \mathbf{A} - \mathbf{Y}||_F^2 + \frac{\gamma}{2} trace(\mathbf{A}^T \hat{\mathbf{L}} \mathbf{A}) \ , \tag{4.26}$$

where $\hat{\mathbf{L}} = \mathbf{Z}^T \cdot \mathbf{L} \cdot \mathbf{Z}$ is an memory-wise and computationally tractable alternative of the Laplacian matrix $\mathbf{L}$. The matrix $\mathbf{A} = [\boldsymbol{a}_1, ..., \boldsymbol{a}_c] \in \mathbb{R}^{m \times c}$ is the soft label matrix for the representative samples, in which each column vector accounts for a class. The matrix $\mathbf{Y} = [\boldsymbol{y}_1, ..., \boldsymbol{y}_c] \in \mathbb{R}^{n \times c}$ a class indicator

matrix on ambiguously labeled samples with $Y_{ij} = 1$ if the label l_i of sample i is equal to j and $Y_{ij} = 0$ otherwise.

In order to calculate the Laplacian matrix $\mathbf{L}$, the adjacency matrix $\mathbf{W}$ needs to be calculated, since $\mathbf{L} = \mathbf{D} - \mathbf{W}$, where $\mathbf{D} \in \mathbb{R}^{n \times n}$ is a diagonal degree matrix such that $\mathbf{D}_{ii} = \sum_{j=1}^{n} \mathbf{W}_{ij}$. In this case $\mathbf{W}$ is approximated as $\mathbf{W} = \mathbf{Z} \cdot \mathbf{\Lambda}^{-1} \cdot \mathbf{Z}^T$, where $\mathbf{\Lambda} \in \mathbb{R}^{m \times m}$ is defined as: $\Lambda_{kk} = \sum_{i=1}^{n} Z_{ik}$. The solution of the Eq.(4.26) has the form of:

$$\mathbf{A}^* = (\mathbf{Z}^T \cdot \mathbf{Z} + \gamma \hat{\mathbf{L}})^{-1} \mathbf{Z}^T \cdot \mathbf{Y} \ . \tag{4.27}$$

Each sample label is, then, given by

$$\hat{l}_i = arg \max_{j \in \{1,...,c\}} \frac{\mathbf{Z}_i \cdot \boldsymbol{\alpha}_j}{\lambda_j} \ , \tag{4.28}$$

where $\mathbf{Z}_{i\cdot} \in \mathbb{R}^{1 \times m}$ denotes the i-th row of $\mathbf{Z}$, and the normalization factor$\lambda_j = \mathbf{1}^T \mathbf{Z} \boldsymbol{\alpha}_j$ balances skewed class distributions.

4.5.5 *Maritime Target Detection*

Having constructed a training set, $S = \{\boldsymbol{f}_i, l_i\}_{i=1}^{n}$, a binary classifier, capable to discriminate pixels that depict some part of a maritime target from pixels that depict the background, can be trained. In this work we choose to utilize Support Vectors Machine (SVM) to transact the classification task.

Let us assume that the classes of negative and positives samples are linear separable. This means that there exists a hyperplane $\mathcal{P} = \boldsymbol{w} \cdot \boldsymbol{f} - b = 0$ that separates the two classes ($\boldsymbol{w}$ is the normal vector to the hyperplane). SVM classifier tries to estimate and maximize the distance between two other hyperplanes, $\mathcal{P}_p = \boldsymbol{w} \cdot \boldsymbol{f} - b = 1$ and $\mathcal{P}_n = \boldsymbol{w} \cdot \boldsymbol{f} - b = -1$, that separate the two classes with no sample existing between them. This can be expressed by the following constraints:

$$\begin{aligned} \boldsymbol{w} \cdot \boldsymbol{f}_i - b \geq 1 \quad \text{if } l_i = 1 \, , \\ \boldsymbol{w} \cdot \boldsymbol{f}_i - b \leq -1 \quad \text{if } l_i = -1 \ . \end{aligned} \tag{4.29}$$

Exploiting the value of labels the pair of constraints in Eq.(4.29) can be rewritten as

$$l_i(\boldsymbol{w} \cdot \boldsymbol{f}_i - b) \geq 1 \quad \text{for } i = 1, ..., n \ . \tag{4.30}$$

The equality of constraint of Eq.(4.30) holds for the samples that lie on the hyperplanes $\mathcal{P}_p$ and $\mathcal{P}_n$. These sumples are called support vectors. The distance between these two hyperplanes is $2/||\boldsymbol{w}||$, which implies that SVM try to solve the following optimization problem:

$$\begin{aligned} &\min_{\bar{w},b} \frac{1}{2} ||\boldsymbol{w}||^2 \\ &\text{s.t.} \quad l_i(\boldsymbol{w} \cdot \boldsymbol{f}_i - b) \geq 1 \quad \text{for } i = 1, ..., n \ . \end{aligned} \tag{4.31}$$

This formulation ensures that the maximum margin classifier classifies each example correctly, which is possible since we assumed that the data is linearly separable. In cases where the two classes are not linearly separable, to allow classification errors, the optimization problem of Eq.(4.31) is transformed to (Cortes and Vapnik, 1995b):

$$\begin{aligned} &\min_{\bar{\boldsymbol{w}},b,\boldsymbol{\xi}} \frac{1}{2} ||\boldsymbol{w}|| + c \sum_{i=1}^{n} \xi_i \\ &\text{s.t.} \quad l_i(\boldsymbol{w} \cdot \boldsymbol{f}_i - b) \geq 1 - \xi_i \quad \text{for } i = 1, ..., n \text{ and } \xi_i \geq 0 \ , \end{aligned} \tag{4.32}$$

where $\xi_i \geq 1$ are variables that allow a sample to be in the margin or to be misclassified and c is a constant that weights these errors.

In the framework of maritime detection, SVM must be able to handle unbalanced classification problems, due to the fact that maritime target usually occupy the minority of captured frames' pixels let alone their total absence from the scene for large time periods. To address this problem, the misclassification error for each class is weighted separately. This means that the total misclassification error of Eq.(4.31) is replaced with two terms:

$$c \sum_{i=1}^{n} \xi_i \rightarrow c_p \sum_{\{i|l_i=1\}} \xi_i + c_n \sum_{\{i|l_i=-1\}} \xi_i \ , \tag{4.33}$$

where c_p and c_n are constant variables that weight separately the misclassification errors for positive and negative examples. The solution of Eq.(4.32) with the classification error of Eq.(4.33) results to a trained SVM, which is capable to classify the pixels of new captured frames.

4.5.6 *Detector Adaptation to New Visual Conditions*

A robust maritime surveillance system must retain high performance for long time periods. Thus, an SVM adaptation mechanism has to be developed, allowing the classifier to be adapted to dynamically changing visual conditions.

Let us denote as $V_1, ..., V_{15}$ the fifteen visual attention maps described in section 4.4. We define the *average* visual attention map, V_{avg}, as

$$V_{avg} = \frac{1}{15} \sum_{i=1}^{15} V_i \ . \tag{4.34}$$

The elements' value of Eq.(4.34) expresses the overall probability a pixel to depict some part of a maritime target. Then, we define the *refined* visual attention map V_r as the outcome of the element-wise multiplication between V_{avg} matrix and background modeling algorithm output B.

Classifier adaptation process is triggered by an automated decision mechanism. Let us define as $V_{r,n}$ and T_n the refined visual attention map and the output of the classifier, respectively, at frame n. When the difference between $V_{r,n}$ and T_n exceeds a predefined threshold the decision mechanism triggers the adaptation process.

During the adaptation process the SVM classifier is retrained. We form a new training set that contains as elements the support vectors of the previously trained classifier, the κ elements of V_r that present the highest probability (positive samples) and have denoted as belonging to the negative class, and the κ elements of V_{avg} that present the lowest probability and have been denoted as background by the background modeling algorithm (negative samples).

Finally, we assume that visual conditions in a maritime environment are smoothly and gradually changing. This implies that the values for $\boldsymbol{w}$, b and $\boldsymbol{\xi}$ of the adapted classifier should be close to the estimated values, $\bar{\boldsymbol{w}}$, $\bar{b}$ and $\bar{\boldsymbol{\xi}}$, of the previously trained classifier. To reduce the time required for classifier retraining, the aforementioned assumption, allows us to speed up the convergence of the optimization algorithm, which seeks for a solution to the problem defined in Eq.(4.32), by restricting the feasible solutions region (set the initial values of the under optimization parameters to the values of $\bar{\boldsymbol{w}}$, $\bar{b}$ and $\bar{\boldsymbol{\xi}}$).

4.6 EXPERIMENTAL RESULTS

The research presented in this work is part of POSEIDON project[1]. Most of the algorithms were developed exclusively in C++ to achieve high performance (as we mention in subsection 4.6.1, the overall system works almost in real time, 17fps, for frames with dimensions 384×288 pixels). There is, also, code in Python, concerning visual attention maps construction, available to download[2]. The performance of each system's component have been checked separately; extracted features were evaluated in terms of discriminative ability and importance, semi-supervised labeling for the predicting outcome and, finally, the binary classifier for its performance.

4.6.1 *Dataset Description*

As mentioned before, our system operates almost in real time, 17fps, for frames with dimensions 384×288 pixels, due to the visual attention maps construction; low level features extraction consists the main computational bottleneck of our system. However, the proposed approach can be expanded for video frames of greater resolution, since each feature can be extracted independently. Thus, feature extraction process can be easily parallelized using multiple processing units (e.g. multiple CPU threads or GPU implementation). Yet, pallelization of this process is out of the scope of this work.

The data sets describe real life scenarios, in various weather conditions. As long as the camera is able to capture a vessel (i.e. spans an area of more than 40 pixels in the frame) the system will likely detect it, regardless the weather conditions (e.g. rain, fog , waves etc.). Apparently, system's performance declines badly in cases of low luminosity due to sensor related sensitivity constraints. Better sensors can partially deal with such issues, but resulting in greater hardware costs.

Data consists of recorded videos from cameras mounted at the Limassol port, Cyprus and Chania Venetian port, Crete, Greece. Monocular cameras were recording videos streams depicting maritime traffic for over one year. Unfortunately, for the vast majority of the video frames, maritime targets are absent from the scene. In order to deal with such cases, we manually edited the videos and kept only the tracks that depict intrusion of one or more targets in the scene. Then, we manually labeled the pixels of key video frames, *keyframes*, to create a ground truth dataset for evaluating our system.

Keyframes originate from raw video frames, on a constant time span equals to t frames i.e. frames that correspond to time instances $t, 2t, 3t, \cdots$ The time span is selected to be 6 seconds, which means that one frame out of 150 is denoted as keyframe. We followed this approach for practical reasons. Firstly, it would be impossible to manually label all video frames at a framerate of 25 fps. Also, the time interval of 6 seconds is small enough to allow the detection of the intrusion of a maritime target in the scene. At this point it has to be clarified that feature extraction task, as well as the binary classification are performed for all frames of a video track. Keyframes are used only for system's performance evaluation.

1 http://www.poseidonproject.gr/
2 https://github.com/kmakantasis/poseidon_features.git

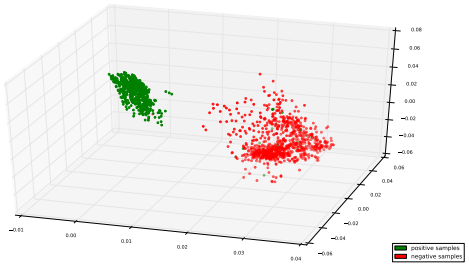


Figure 4.5: Positive and negative samples plotted in 3-dimensional space. PCA was used to extract the 3 dominant components of the dataset. The two classes are linearly separable.

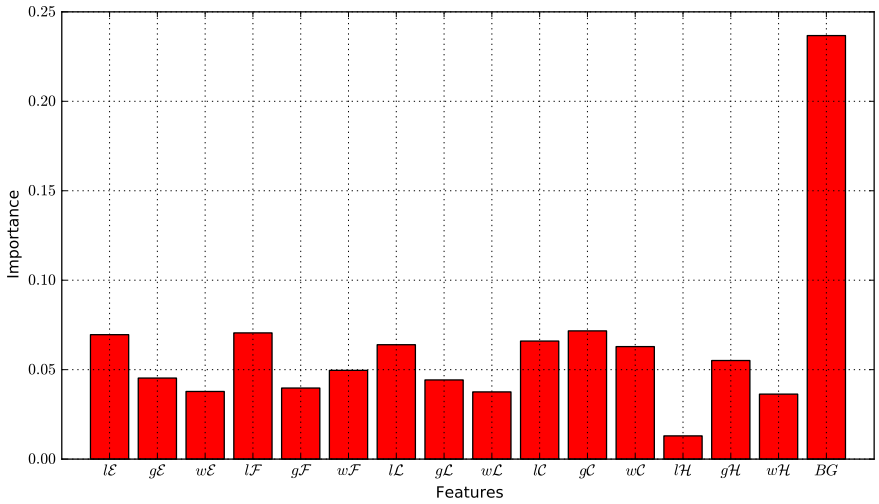


Figure 4.6: Features importances. The feature that corresponds to output of background modeling algorithm, which implicitly captures the presence of motion in the scene, is presented to be he most important.The rest of the features contribute almost the same to the classification task, except from the feature that corresponds to the local descriptor of image entropy, which presents the lowest importance.

4.6.2 *Evaluation of Extracted Features*

In this section, we examine if the extracted features are able to describe appropriately frame's pixels and, consequently, provide useful information that will facilitate the classification task. In addition, the extent that each one of the features affects the classification task (i.e. how important a feature is) is examined. Results concerning the importance of features, may allow us to discard some of them, in order to speed up system's performance.

To evaluate features information, we utilized the keyframes' ground truth data. The feature extraction task results in a 16-dimensional feature vector for each pixel in a frame. The quality of features' information is evaluated through dimensionality reduction and samples plotting, in order to visually examine their distribution in space, see Figure(4.5). The two classes, as shown in Figure(4.5), are linearly separable, which suggests high quality features. The small amount of positive samples, that lie inside the region of the negative class, correspond to maritime targets' contours and probably occurred due to segmentation errors during manual labeling.

Except for the evaluation of the constructed vector as a whole, the importance of each one of the extracted features is examined separately, in order to define how much each one of the features affects the classification task. The importance of features is specified via Forest of Randomized Trees (FRT). FRT is a non parametric supervised learning method, whose goal is to create a model that predicts the value of a target variable by learning simple decision rules, inferred from the data features. Each node of the trees codes a decision rule, which is expressed as comparison of the value of a specific feature with a constant threshold.

The relative rank (i.e. depth) of a feature used as a decision node in a tree can be used to assess the relative importance of that feature with respect to the predictability of the target variable. Features used at the top of the tree contribute to the final prediction decision of a larger fraction of the input samples. The expected fraction of the samples they contribute to can thus be used as an estimate of the relative importance of the features.

In Figure(4.6) the relative importance of each one of the extracted features is presented (features labeling follows the notation of Section 4.4). The dominant feature is the one that corresponds to the output of background modeling algorithm, which, in practice, captures the presence of motion in the scene. The rest of the features contribute almost the same, except from the feature that corresponds to the local descriptor of image entropy. This is mainly caused by the fact that entropy feature of a specific pixel is calculated taking into consideration only its neighboring pixels, emphasizing this way pixels that are a little bit different than their neighbors, such as pixels that actually depict small sea waves or clouds in the sky. However, the importances of global and window descriptors, which are based on the local descriptor, for the feature of entropy, suggest that the local entropy information can be effectively used to facilitate classification task.

Our algorithm overcomes possible drawbacks of other related approaches. Firstly, the combination of low-level image features with foreground extraction techniques allows our systems to operate with active cameras, something which is not possible for techniques based only on background subtraction (Kaimakis and Tsapatsoulis, 2013). Secondly, contrary to (Socek et al., 2005), our system exploits not only color information but structural knowledge as well. Finally, the proposed system extends the work of (Makantasis et al., 2013) by exploiting entropy and frequency features, in order to increase its robustness.

4.6.3 *Evaluation of Semi-supervised Labeling*

In order to evaluate semi-supervised labeling, we assume that manual labeling of keyframes contains no segmentation errors. The ratio of the representatives samples in relation with the ambiguously labeled samples is the only factor that affect the performance of labeling algorithm.

As shown in Figure(4.7), the labeling error is lower than 2% when the ratio of the representatives samples in relation with the ambiguously labeled samples is over 40%. When the ratio is smaller than 40% the labeling error is linearly increasing and it reaches the value of 5.7% when the ratio of representative samples is 10%.

The choice for an appropriate value for the ratio of representatives is inherently dependent on the quality of human based labeling. If labeling is the result of a rough image segmentation, a lot of the labeled pixel will

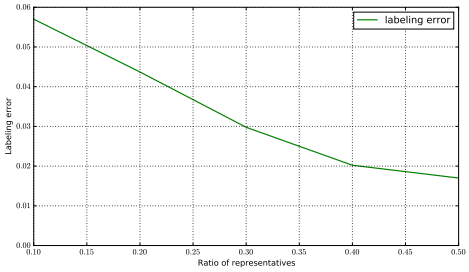


Figure 4.7: Semi-supervised labeling performance. When ratio of representative samples is over 40% the labeling error is lower than 2%.

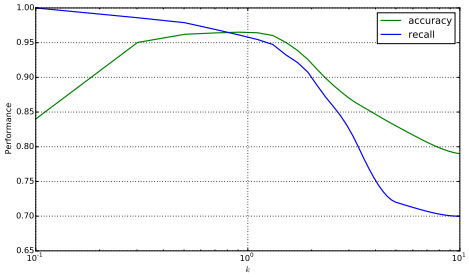


Figure 4.8: The recall of the system is inversely proportional to k. The penalties for misclassifying positive and negative samples are inversely proportional to the cardinalities of their classes.

carry the wrong label. In such cases the aforementioned ratio must be set to a small value. The most representative samples from each class is assumed that carry the right label, while the labels of the rest of the samples must be reconsidered.

In our case, we required the user to segment the frame in a very careful way, which implies that the vast majority of the pixels will carry the right label. For this reason we set the ratio value to 40%. The semi-supervised labeling algorithm with 40% of representatives is expected to re-label 1.7% of the samples.

4.6.4 *Binary Classifier Evaluation*

The performance of the binary classifier is dependent on the values of the parameters c_p and c_n of Eq.(4.33). Let us denote as n_p and n_n the number of samples in positive and negative class respectively. To examine the influence of parameters c_p and c_n on classification accuracy we define the parameter

$$k = \frac{c_n \cdot n_n}{c_p \cdot n_p} \ . \tag{4.35}$$

In practice, parameter k assigns different weights to misclassification errors, which correspond to positive and negative examples. When the value of k is equal to one, the weights that penalize misclassification sample for each class are inversely proportional to the cardinalities of the classes. When $k < 1$ a bigger penalty is assigned to false negatives, while for $k > 1$ false positives are considered more important. False negatives correspond to pixels that actually depict some part of a maritime target, but are denoted as background by the classifier.

However, a maritime surveillance system must emphasize on minimizing the false negative rate. In other words, it is more important, the system to detect all potential maritime targets, even if it will raise a small amount of false alarms, than minimizing false positives at the cost of missing target intrusions.

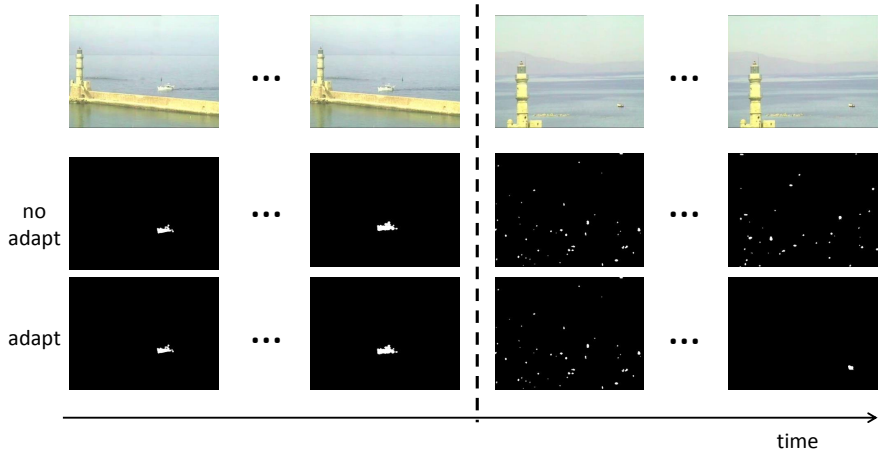


Figure 4.9: Adaptation mechanism. The dotted line represent the time the scene changes. Before that time both classifiers performs the same. While at the time the scene changes the performance of both classifiers collapses, the one that exploits the adaptation mechanism adapts its operation to new visual conditions and achieves high performance in the following frames.

Figure(4.8) presents the performance of classifier for different values of parameter k. The green line represents classification accuracy, while the blue line the recall of the system. If we denote as p_c the set of pixel that denoted by the classifier as positive samples and as p_t the set of pixels that actually belong to the positive class, then recall ρ is defined as

$$\rho = \frac{p_c \cap p_t}{p_t} \ . \tag{4.36}$$

When ρ is equal to one, all true positive samples have been correctly classified by the binary classifier. Accuracy is the proportion of correctly classified samples of the whole dataset. As shown by the green line in Figure(4.8) the accuracy of the classifier reaches its maximum value, when k is equal to one. On the other hand, the recall of the system is monotonically decreasing as the value of k is increasing. In our case we set $k = 0.7$ to balance between maximizing classification accuracy and minimizing false negative rate. For $k = 0.7$ the accuracy of the classifier is equal to 96.4%, while recall is equal to 97.1%.

Finally, in Figure(4.9) the importance of the SVM updating mechanism is illustrated. Two different video sequences were concatenated for evaluating the updating procedure. The two video sequences were depicting the same scene under a different perspective and were captured at different time of day. Variations in perspective affect window and global descriptors, while the time of recording affects illumination conditions, and thus the values of all descriptors.

We evaluate the performance of two classifiers; one that supports the adaptation mechanism and one who does not. The vertical dotted line in Figure(4.9) denotes the time instant that the scene changes. Before that time both classifiers perform the same. At the time of a scene change, the performance of both classifiers collapses. In the following frames, the operation of the classifier that exploits the adaptation mechanism is capable to adapt to the new visual conditions, improving its performance. On the contrary, the

performance of the non adaptive classifier continues to remain low. By setting the value of κ equal to 100 (see subsection 4.5.6) the adaptation process takes less than one second (15 frames at 25fps) to be completed.

4.7 CONCLUSIONS

A vision based system, using monocular camera data, is presented in this work. The system provides robust results by combining supervised and unsupervised methods, appropriate for maritime surveillance, utilizing an innovative initialization procedure. The system offline initialization is achieved through a graph based SSL algorithm, suitable for large data sets, supporting users during segmentation process. Another advantage is the automated adaptation of the system to new environments, in real time.

Extensive performance analysis suggest that the proposed system performs well, in real time, for long periods without any special hardware requirements and without any assumptions related to scene, environment and/or visual conditions. Such system is expected to significantly support the local authorities, or anyone interested in maritime surveillance without any significant additional cost.

5

VISION BASED ACTIVITY RECOGNITION IN INDUSTRIAL WORKFLOW

5.1 MOTIVATION

Recognizing human activities and behaviors in real-world environment finds application in a variety of domains including virtual reality, human-computer interaction and smart video surveillance. Especially when it comes to smart monitoring of large scale industrial environments, the importance of activity recognition highly relates to the safety and security of the employees, to the reduction of costs and the optimization of production scheduling.

5.2 RELATED WORK

Accurate activity recognition is a highly challenging task due to the diversity of the activities and types of behaviors to be recognized let alone cluttered backgrounds, occlusions and viewpoint variations. Therefore, most of the existing work (Efros et al., 2003; Jhuang et al., 2007; Laptev and Perez, 2007; Veres et al., 2011; Kosmopoulos et al., 2012; Protopapadakis et al., 2013) follows the typical paradigm of pattern recognition, which consists of two separate steps; firstly, the computation of complex handcrafted features by making certain application depended assumptions, and secondly, learning classifiers based on the obtained features. However, in real-world scenarios, it is rarely known which features are important for the task at hand. Especially for human activity recognition and behavior understanding, different action classes may appear complete different in terms of their appearances.

Contrary to approaches that rely on handcrafted features, deep learning models (Lecun et al., 1998; Hinton and Salakhutdinov, 2006a; Salakhutdinov and Hinton, 2009a; Lee et al., 2009) can learn complex inputs representations by building high-level features from low-level ones, automating the process of feature construction. A special type of deep models are the Convolutional Neural Networks (CNNs). CNNs alternatively apply trainable filters and pooling operations on 2D inputs resulting in a hierarchy of increasing complex features.

Human activities are time varying processes occurred in a sequence of frames and, thus, described by 3D features, i.e. spatial (2D) information that is present in the visual content of each frame, plus temporal information encoded in a sequence of frames. Therefore, utilization of CNNs for human activity recognition and behavior understanding requires either the application of the computationally expensive 3D convolution on raw data (Ji et al., 2013), in order to take into consideration the temporal dimension, or the appropriate transformation of each frame's visual content to incorporate temporal information into each one of the frames.

In order to avoid the high computational cost of 3D convolution, we transform the raw data to fuse spatio-temporal into each frame visual content. Specifically, we encode each frame's information by constructing the Motion History Image (MHI) using a predefined number of precedent frames. It has been shown in (Schindler and Van Gool, 2008) that a small number of

subsequent frames (5 to 7 frames) are enough to achieve an activity recognition performance similar to the one obtainable with the entire video sequence. The MHI of each frame is fed as input into a CNN to hierarchically construct complex features, which, then, are fed as input into a Multi-Layer Perceptron (MPL) in order to classify the frame under consideration into a predefined number of activity classes. Using the aforementioned approach, our system (i) hierarchically constructs spatio-temporal features avoiding the high computational cost of 3D convolution, and (ii) achieves real-time predictions due to the feed-forward nature of CNNs and MLPs.

5.3 PROPOSED METHODOLOGY

We consider the problem of human activity recognition in industrial workflows. A workflow is a process that happens repetitively and consists of a sequence of discrete tasks. The definition of tasks stems from domain knowledge and each task spans a time interval and is described by a set of sequential frames. Therefore, the problem of human activity recognition can be seen as the classification of each frame to one of the available tasks (classes).

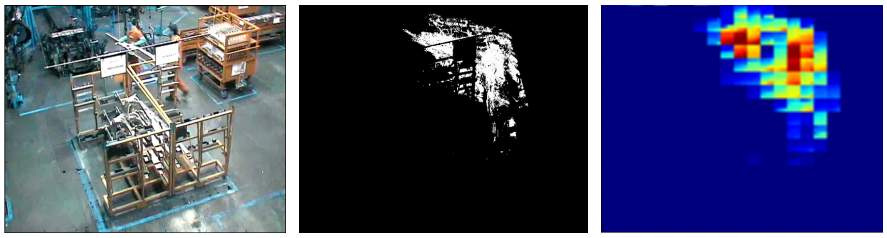
Classification task requires the description of each frame by a set of features that fuse spatial and temporal information. However, construction of handcrafted features is inherently depended on the problem at hand. To overcome this limitation, we exploit a deep learning architecture to automatically construct high-level features. Feature construction using raw video frames takes into consideration the spatial information. However, such an approach does not consider the temporal information encoded in multiple contiguous frames. In order to incorporate temporal information into constructed features, we model the visual content of each frame using its MHI, which captures the history of a task that is being executed.

CNNs apply trainable filters and pooling operations on their input, resulting in a hierarchy of increasingly complex features. Convolving their input with the trainable filters consists the main bottleneck during training and prediction phases. In order to achieve a computational efficient architecture, we split MHI input of CNN into non overlapping windows and keep the most dominant Discrete Cosine Transform (DCT) coefficients. This way, we are able to reduce the size of the input, and thus decrease computational cost, while at the same time we achieve minimal information loss.

The output layer of the CNN in sequentially connected with a MLP, which carries out the classification task. We choose to use a MLP due to its global function approximation properties. The unified learning architecture is trained in a supervised manner using the well-known backpropagation algorithm.

5.4 TASK MODELING

In the following we describe how we represent the tasks. In subsection 5.4.1 we present how MHI is computed for each frame, while in subsection 5.4.2 we describe the dimensionality reduction of CNN input.



(a) (b) (c)

Figure 5.1: Task modeling overview. (a) Original captured frame, (b) MHI for the captured frame and (c) dimension reduction of the captured frame using DCT transform.

5.4.1 *Visual Observations*

We encode spatial and temporal information of visual observations using the MHI. MHI for a specific frame is computed using a sequence of, let's say d, precedent frames. For each precedent frame silhouette mask that has non-zero pixels where the motion occurs is created. Let us denote as f_t the number of frame at time t for which we want to compute the MHI and as $s(x, y)$ the pixel value for silhouette mask at location (x, y). Then, according to (Davis, 2001a), the MHI, $m_t(x, y)$, for a pixel located at the same position for frame f_t is computed as

$$m_t(x,y) = \begin{cases} f_t & \text{if } s(x,y) \neq 0 \\ 0 & \text{if } s(x,y) = 0 \\ & \text{and } m_{t-1}(x,y) < f_t - d \\ m_{t-1}(x,y) & \text{otherwise} \end{cases} . \tag{5.1}$$

The MHI captures the essence of the underlying motion pattern in a scene; in our case motion pattern of a human activity. Both where the motion is happening and also how the motion is occurring are present in one compact template representation. In an MHI, pixel intensity is a function of the motion history at that location, where brighter values correspond to more recent motion.

Using the above definition of MHI it is easy to be seen that the values of m_t are highly related to the ordinal number of frame under consideration. To facilitate the process of learning architecture training, we use a normalized version of MHI, whose values are kept into a specific range. Specifically, if we denote as μ_t the normalized version of m_t, then $\mu_t(x, y)$ is computed as

$$\mu_t(x,y) = \frac{m_t(x,y) - \min\{m_t^+\} + 1}{d} \tag{5.2}$$

where m_t^+ is the set that contains all positive elements of m_t.

5.4.2 *Dimension Reduction of the CNN Input*

CNNs apply trainable filters and pooling operations on their input. However, convolving their input with the trainable filters consists the main bottleneck during training and prediction phases. In order to achieve a computational efficient framework the dimension of the input must be reduced.

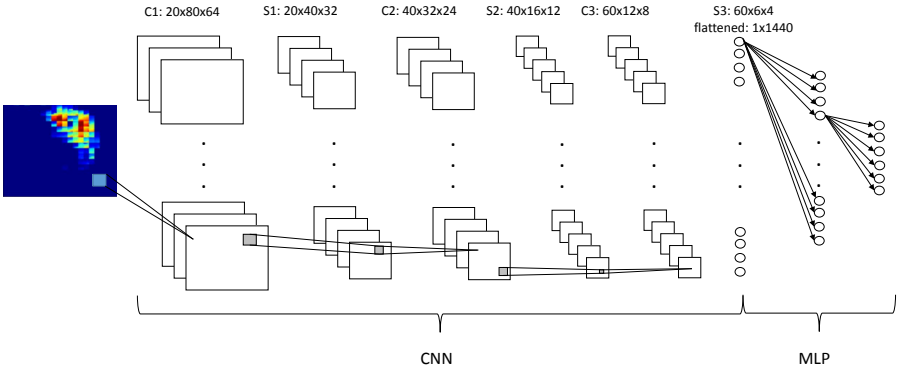


Figure 5.2: Overall architecture of the learning model. C1, C2, C3, S1, S2 and S3 correspond to the three convolutional and max pooling layers respectively.

The most common approach for reducing the dimension of an image is through subsampling operations. However, such approaches may discard valuable information. Therefore, we choose to borrow image compression means to reduce the dimension of the input. Concretely, we choose to use 2D DCT, which is a widely used technique for lossy image compression, e.g. JPEG.

In particular, we split the MHI of each frame into 32×32 non-overlapping blocks. For each block we compute the DCT transform and keep its most dominant coefficients. In our case, we keep the 16 dominant DCT coefficients to encode the visual information of each block. Using 16 coefficients we achieve very low information loss, due to the fact that MHI are grayscale images whose the vast majority of pixels are equal to zero (no motion). This way, each block is described by a 16 dimensional vector, which reshaped to a 4×4 matrix in order to preserve 2D input. Following the aforementioned approach, each 32×32 block is represented by a 4×4 matrix, reducing 64 times the dimension of the original input and at the same time preserving the most important information. The task modeling overview is presented in Fig. 5.1.

5.5 LEARNING MODEL ARCHITECTURE

As it has been mentioned before, CNNs apply trainable filters and pooling operations on their input, resulting in a hierarchy of increasingly complex features. Convolutional Layers (CL) consist of a rectangular grid of neurons (filters), each of which takes inputs from a rectangular section of the previous layer. For reducing the number of network's free parameters the weights for this rectangular section are the same for each neuron in the convolutional layer. Each convolution layer is followed by a Pooling Layer (PL). This layer subsamples block-wise the output of the precedent CL and produces a single output for each block.

Specifically, if we denote the k-th output of a given CL as h^k whose filters are determined by the weights W^k and bias b^k then the h^k is obtained by

$$h_{ij}^k = g((W^k * x)_{ij} + b^k) \ , \tag{5.3}$$

where x stands for the input of the CL and indices i and j correspond to the location of the input where the filter is applied, (*) stands for the convolution

operator and $g(\cdot)$ is a non-linear function. PLs simply take some $k \times k$ region and output the maximum value in that region, i.e. if their input layer is a $N \times N$ matrix, they will then output a $N/k \times N/k$ matrix.

5.5.1 *Deep Learning Model Parameterization*

During the validation of our model, we used some very challenging videos from the production line of a major automobile manufacturer (see (Voulodimos et al., 2011)). The size of video frames for this dataset is 704×576 pixels. Following the procedure described in section 5.4 spatial and temporal information for each frame is represented by a 88×72 matrix, which is fed as input to the first CL of our learning model.

The input is convolved with 20 filters of size 9×9 and the output of this layer, a 3D matrix of dimension $20 \times 80 \times 64$ is fed to the first PL, which outputs a 3D matrix of dimension $20 \times 40 \times 32$. The first CL consists of 1640 trainable weights. The output of the first PL is fed to the second CL and convolved with 40 trainable filters of dimension 9×9. The output of the second CL is fed to the second PL, which outputs a 3D matrix of dimension $40 \times 16 \times 12$. The second CL consists of 3280 trainable weights. The output of the second PL is fed to the third CL and convolved with 60 trainable filters of dimension 5×5. Again, the output of this layer is fed to the following PL, which outputs a 3D matrix of dimension $60 \times 6 \times 4$. The third CL consists of 1560 trainable weights. The output of the last PL is flattened to form a 1440 dimensional feature vector, which is fed as input to the MLP. The MLP contains one hidden layer with 600 neurons and an output layer with 6 neurons (tasks are classified to 6 different classes). MLP contains 868206 trainable weights (see Fig.5.2).

5.6 EXPERIMENTAL VALIDATION

Validation of our model took place using some very challenging videos from the production line of a major automobile manufacturer (Voulodimos et al., 2011). The production cycle includes tasks of picking several parts from racks and placing them on a designated cell some meters away (see Fig5.3). Each of these tasks is regarded as a class of behavioral patterns that have to be recognized. Although, video frames have been divided into 7 different classes, the 7-th class is a null class (workers are idle or absent) and thus frames belonging to 7-th class were not considered for classification purposes.

5.6.1 *Experimentation Setup*

We developed our learning model using Theano (Bastien et al., 2012; Bergstra et al., 2010) library in python. The model was built on a conventional laptop with i7 CPU and 8GB RAM. The considered dataset consists of 20 scenarios. We compared the performance of our system to the performance of state-of-the-art techniques. Specifically, the first method that we used was Echo State Networks (ESN) (Veres et al., 2011), which exploits handcrafted local motion grid features. Furthermore, we compared the performance of our method to Hidden Markov Model exploiting Particle Filters (HMM-PF) and Hidden Markov Model using a Neural Network rectification scheme (HMM-

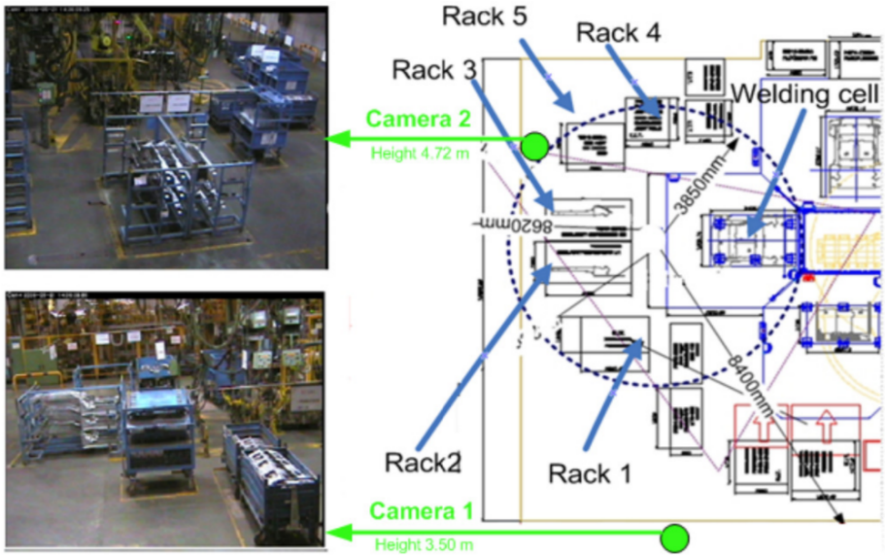


Figure 5.3: Depiction of the work cell along with the position of camera 1 and the racks.

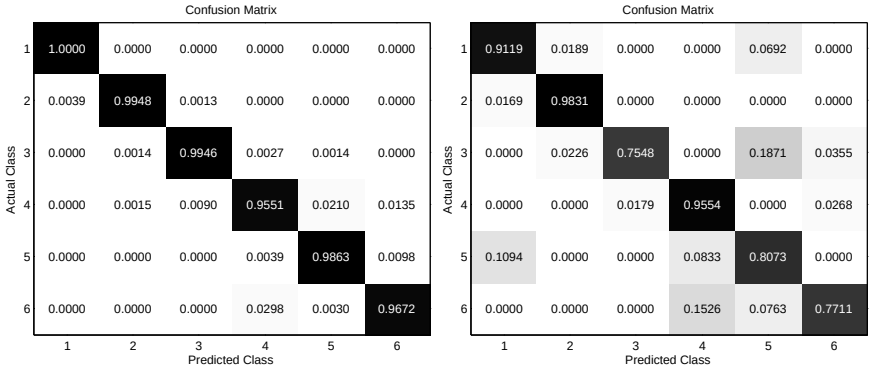
NN) (Kosmopoulos et al., 2012). These methods are also rely on handcrafted features obtained using Zernike moments of pixel changing history (Xiang and Gong, 2006). Finally, we evaluated classification performance against MultiClass Tapped Delay Support Vector Machines (MC-TDSVM) (Protopapadakis et al., 2013). This technique exploits the same handcrafted features as in (Kosmopoulos et al., 2012) along with a user feedback strategy.

In contrast to these approaches, where all 20 scenarios were used to train the learning models, we make things even harder by using 15 scenarios to create training, validation and testing sets and keeping the remaining 5 scenarios to test the generalization ability of our system to completely unknown data. The dataset containing the frames of the 15 scenarios was divided into three sets, i.e. training, validation and testing data with split ratio 7 : 1.5 : 1.5. That is, we randomly choose 70% of the whole dataset as the training set, and 15% and 15% for the validation and testing sets.

5.6.2 *Results*

Using the aforementioned experimental setup, we created the confusion matrices to quantitatively evaluate the performance of our model (see Fig.5.4). Furthermore, we computed average precision and recall over all classes (see Table5.1).

The quantitative performance evaluation in terms of average precision and recall, presented in Table5.1, shows that our method outperforms all other techniques. We should note that ESN, HMM and SVM methods were using data from all 20 scenarios, while for our method only 15 scenarios were used. Concerning performance on testing set, our method improves classification accuracy over 10% compared to HMM-NN and over 20% compared to ESN. Generalization capability of our deep learning model is presented in the last row of Table5.1. Our method performs almost the same with HMM-NN and MC-TDSVM and better than all other techniques, despite the fact that it is applied on completely unknown data (data form



(a) (b)

Figure 5.4: Confusion matrices presenting the performance of our system for each class. Classification accuracy (a) on the testing set and (b) on completely unknown data.

Method	**precision**	**recall**
Echo State Networks	0.777	0.772
HMM-PF	0.797	0.788
MC-TDSVM (40% training set)	0.857	0.857
HMM-NN	0.875	0.863
MC-TDSVM (60% training set)	0.871	0.863
Our Approach (testing set)	**0.983**	**0.978**
Our Approach (unknown data)	**0.867**	**0.892**

Table 5.1: Quantitative evaluation results. Performance comparison against state-of-the-art techniques.

the remaining 5 scenarios), which does not stand for other techniques case (training set contains data from all scenarios).

Fig.5.4(a) presents proposed method's performance on testing set for each class in the form of a confusion matrix, while Fig.5.4(b) presents the classification accuracy of our system on completely unknown data form the remaining 5 scenarios. Misclassification errors, concerning the testing set, is lower than 4.5% for all classes. Such errors may occurred due to cluttered background, occlusions and image sensor noise.

Concerning the performance on completely unknown data, Fig.5.4(b), we see that misclassification rate for the 1st and 2nd classes are very low, 8.8% and 1.3% respectively. However, 18% of the samples that actually belong to the 3rd task have been assigned to the 5th class and 8% and 15% of the samples that actually belong to 5th and 6th classes have been assigned to the 4th class. This mainly happens due to the similarity of motion patterns presented in tasks 3 to 6. Motion patterns are encoded in MHIs, which then are used, by our learning model, to hierarchically construct features for discriminating human activities. Therefore, similar motion patterns for different tasks can increase misclassification rate. However, as shown in Fig.5.4(a), our learning model has the capacity to discriminate, even very similar ex-

amples that belong to different classes, if it is trained using a highly representative dataset.

5.7 CONCLUSIONS

We proposed a deep learning based approach for human activity and behavior understanding in industrial environments. Our system exploits CNNs to hierarchically construct complex features and a MLP to carry out the classification task. The validation of our model was conducted using a very challenging dataset from the production line of a major automobile manufacturer. Experimental results show the superior performance of our system among state-of-the-art techniques.

Part II

FROM UNSTRUCTURED VISUAL CONTENT TO OBJECTS

In this part, we investigate how the visual content that is stored that is stored in distributed and heterogeneous Internet databases can be, initially, organized, and then utilized towards objects documentation. There are two different chapters, where we try to address the problem of retrieving, based on visual content, and dynamically indexing user generated photographs available over the web.

6 UNSTRUCTURED VISUAL CONTENT

6.1 THE NOTION OF UNSTRUCTURED DATA

Unstructured data refers to information that either does not have a predefined data model or is not organized in a predefined manner. This results in irregularities and ambiguities that make it difficult to understand using traditional programs as compared to data stored in fielded form in databases or annotated (semantically tagged) in documents.

Software that creates machine-processable structure exploits the linguistic, auditory, and visual structure inherent in all forms of human communication. Algorithms can infer this inherent structure from text, for instance, by examining word morphology, sentence syntax, and other small- and large-scale patterns. Unstructured information can then be enriched and tagged to address ambiguities and relevancy-based techniques then used to facilitate search and discovery.

In the context of visual information such software usually referred as Content Based Image Retrieval (CBIR) system. Such a system relies on the visual content of images and/or videos in order to organize unstructured visual data. Towards this direction, CBIR tools that mine relevant images from large repositories mainly use image filtering and clustering algorithms to appropriately organize image data into groups of similar visual properties. Thus, CBIR methods can be considered as suitable tools towards efficient visual content-based image filtering; especially those ones that are mostly focused on content organization and image matching.

In the following chapters of this part we describe two different scenarios; the first one focus on the development of a fully automatic approach for content-based filtering of Internet stored data used for cultural heritage applications, while the second one presents the development of an online image indexing structure. For both scenarios we use existing related works for CBIR systems as a stepping stone for developing a state-of-the-art CBIR, filtering and indexing system, applicable to cultural heritage e-documentation, that is capable to organize unstructured image data and furthermore discover those images that are the most appropriate for 3D reconstruction.

7

IN THE WILD IMAGE RETRIEVAL AND CLUSTERING

7.1 MOTIVATION

During the Internet era, there are extremely large collections of images and videos available over distributed web repositories (e.g., Flickr, Picasa, Photosynth). In 2011 Pingdom enterprise reported that 4.5 millions of photos uploaded to Flickr every day[1]. These images cover not only personal events but also historic incidents and cultural heritage assets. Although such proliferation of millions of shared photographs, which are online available for free, provides a unique opportunity for cultural heritage e-documentation, there are limited technological tools and research methods for retrieving, mining and ultimately exploiting such wide cultural heritage collections for 3D reconstruction applications.

One of the most common approach for cultural heritage e-documentation is based on 3D scanning (Barone et al., 2012). In (Karaszewski et al., 2012), a fully automatic approach for 3D measurements is presented as regards preservation of cultural heritage artifacts. In addition, the work of (Sitnik and Karaszewski, 2010) develops a software to cope with very large data volumes obtained using 3D scanning. Finally, examples of creating high resolution 3D volumetric maps for e-documentation of cultural heritage assets are presented in (Bunsch et al., 2012).

In contrast to the aforementioned approaches, where 3D data acquisition is accomplished in a very constrained environment, using specialized equipment, Web based collections ("wild image collections") can be exploited for cultural heritage e-documentation. The main, however, difficulty of using "wild image collections" is that the Internet stored image content is unstructured, requiring new tools in the area of content-based filtering. Consider, for example, a query containing the keywords "Acropolis, Parthenon". As a response to that query, a large set of images are retrieved, which depict not only the Parthenon monument itself, but also the view of the city of Athens from the Acropolis hill. These image outliers confuse any e-documentation algorithm. Although auto-generated geo-location tags can improve visual content characterization and therefore the retrieval performance, they suffer from low precision since geo-information does not correctly describe what is actually depicted. Therefore, *content-based filtering* algorithms are necessary for an effective and computationally efficient 3D reconstruction exploiting distributed Web based image collections. Content-based filtering algorithms, apart from discarding image outliers, also organize the retrieved unstructured content into well-structured forms to optimize both 3D reconstruction performance and computational cost.

Our research exploits User Generated Content (UGC) image collections, stored on distributed multimedia platforms, such as Flickr and Picasa. The content-based filtering engine is developed under the framework of a European research initiative, called "4D-CH-World: Four Dimensional Cultural Heritage World"[2], with the purpose of dynamically creating 3D/4D

1 http://royal.pingdom.com/2012/01/17/internet-2011-in-numbers/
2 http://www.4d-ch-world.eu

reconstructions of cultural heritage objects from "wild Internet image collections" (that is, collection from the Web). 4D-Ch-World integrates and validates a multi-disciplinary research agenda of computer vision, multimedia, databases and photogrammetry tools.

However, the main difficulty in implementing a precise 3D/4D reconstruction of an object from unstructured Internet image collections, (being captured for personal use instead of reconstruction purposes), is that there are several outliers in the set of retrieved data deteriorating both performance and computational cost. While there exists 3D reconstruction algorithms, such as the structure from motion (Wu et al., 2011a, 2012, 2011b), which present robustness against noisy data, their computational complexity significantly increases with respect to the number of input data. This makes direct implementation of such methods for large image volumes practically impossible.

To address this difficulty, we propose an efficient content-based filtering method that operates on two directions; first it filters out (discards) image outliers, that is it excludes images whose visual content is quite dissimilar to the majority of the data (which are considered as relevant) and second it clusters the relevant images into different groups to optimize the performance of the 3D reconstruction engine.

7.2 LITERATURE REVIEW

In a Content Based Image Retrieval (CBIR) scheme, users submit queries to the system and the system responses relevant visual retrievals. The query image acts as a reference image whose visual information is encoded, using, for example a fuzzy representation (Doulamis et al., 2000). Then, images that present high visual similarity with respect to the reference one (i.e., the query image) are retrieved as the most relevant. Towards this direction, Murthy *et al.* (Murthy et al., 2010) proposes a two stage image retrieval process based on the color properties of the reference (query) image. Starting from an initial retrieved image set (first stage), a hierarchical clustering is applied to filter out image retrievals (second stage) to increase matching performance. However, the efficiency of the aforementioned approach inherently depends on the color properties of the reference image and therefore on the shutter speed of the camera, lens aperture and environmental lighting conditions of the scene at the time the photo was taken. Based on the same concept, Chum *et al.* in (Chum et al., 2007) present a system, with the objective to retrieve all views (instances) of an object in a large database upon a query. To achieve this, the authors exploit, apart from visual similarities, a vocabulary tree for indexing and query expansion. Similar to the previous approaches, the system presented by Philbin *et al.* in (Philbin et al., 2007) enables user to select regions of interest within the submitted image query and then the system returns a ranked list of images that contain the selected object. To improve computational efficiency, the research of (Philbin et al., 2007) exploits a fast spatial matching algorithm. Video content based retrieval have been introduced in (Kosmopoulos et al., 2009; Halkos et al., 2009), while a mobile agent is presented in (Papadakis et al., 2008). The main drawback of the aforementioned approaches compared to our method is that they require a reference (query) image or object to carry out the retrieval process. On the contrary, our method eliminates image outliers and

organizes the retrieved results into representative groups under an unsupervised framework.

In this context, Kekre *et al.* in (Kekre et al., 2011) use image signatures, extracted by the color image properties, to create clusters which are then represented by codebooks and stored in a database. Each new query image is compared against the existing codebooks according to its color features in order to estimate the most relevant visual matching. Similar to this approach, the work of (Min and Cheng, 2009) proposes the dominant color descriptor to encode visual information, while clustering is performed using fuzzy Support Vectors Machines (fSVMs). The retrieved set of images is further refined by a user relevance feedback system (Doulamis and Doulamis, 2006, 2004; Doulamis et al., 2003). Additionally, crowd-sourcing methods have been investigated for refining the retrieval results (Ntalianis et al., 2010). However, the performance of the work in (Min and Cheng, 2009) depends on illumination conditions since color descriptors are used to encode visual information as in (Murthy et al., 2010). Simon *et al.* in (Simon et al., 2007) focus on visual clustering implemented through an optimization approach that selects a number of canonical image views for constructing a scene summary. The main limitations of these approaches are that they use global image features to encode visual content. Thus, these approaches are not suitable for cultural heritage applications and especially for 3D reconstruction purposes where we need to select different object views, laying in the spherical coordinates that surround this object, instead of using images that they present quite similar content. Global visual representation fail to describe the different view instances of an object since both the geometry of the foreground as projected onto the 2D image plane and the content of the background are quite dissimilar.

Besides this, textual or geo-location information is exploited to filter out the retrieved results. Towards this direction, the works of (Arampatzis et al., 2013) and (Papadopoulos et al., 2010) exploit geo-tagging and annotation to improve the retrieval performance. Particularly, the work of (Papadopoulos et al., 2010) describes an image analysis algorithm that automates the detection of landmarks and events from large multimedia databases in order to improve content-consumption experience. The idea of geo-clustering is also exploited by the work of (Zheng et al., 2009) for retrieving landmark images. The engine combines geo-information along with a hierarchical agglomerative visual clustering to obtain dense groups. In particular, in the first stage of the algorithm, images that share the same geo-tagging information are extracted. However, the retrieved set contains a lot of image outliers that are photos of several adjacent landmarks. To eliminate the noisy images, visual clustering is proposed as a second stage of the algorithm of (Zheng et al., 2009). The authors in (Agarwal et al., 2009) use geo-tagged datasets from Flickr and assume multiple different views of the same scene in each of these datasets. Then, they create a vocabulary tree and use it for indexing, and query expansion like in (Chum et al., 2007) to cluster together similar images. Again, the main limitation of the aforementioned approaches is that they exploit global visual information to detect the true images of a landmark against the noisy one. Although such approaches are useful for content-based image retrieval applications, where the aim is to extract similar images upon a query, they present many shortcomings when they apply for 3D reconstruction scenarios.

7.3 APPROACH OVERVIEW

We aim at describing a fully automatic approach for content-based filtering of Internet stored data used for cultural heritage applications. The filtering results are then exploited by 3D reconstruction algorithms, like the structure from motion scheme (Wu et al., 2011a, 2012, 2011b) in order to achieve e-documentation of an object or an archaeological site. Instead of the aforementioned state of the art methods, our technique combines a local visual content representation schema with textual and geo-tagging information. The adoption of a local visual representation targets our main objective to select a set of images that are suitable for 3D reconstruction and therefore it contains the same foreground cultural object at different geometric perspectives. In particular, in our research, the ORB (Oriented FAST and Rotated BRIEF) is adopted as local descriptor (Rublee et al., 2011). ORB is a combination of the well-known FAST key-point detector (Rosten and Drummond, 2006) and the recently developed BRIEF descriptor (Calonder et al., 2010), being rotation invariant and resistant to noise. Compared to conventional SIFT (Lowe, 2004) and SURF (Bay et al., 2006), ORB keeps the high performance and robustness of SIFT, while simultaneously being two order of magnitude faster.

The ORB local descriptor is adopted to capture the different geometric perspectives of an object, requiring for a 3D reconstruction. Then, a two way pairwise descriptor matching is applied onto all images sharing the same textual and geo-location information. This way, we construct a similarity matrix that indicates how close the visual content of two images is. Since pair-wise matching is computed through local visual descriptors, we are able to model the different views of the cultural heritage object as projected onto the 2D image plane.

In order to remove the image outliers from the retrieved image set under an unsupervised framework, each image is considered as a point onto a multi-dimensional hyperspace manifold. The coordinates of each image point onto this manifold express the position of the images on the hyperspace and therefore they constitute a clear indicator of how close the images are. The distribution of the retrieved images sharing the same textual and geo-location information onto the multi-dimensional manifold is expected to form a) a compact hyperspace on which images depicting different views of a cultural object are located and b) space samples of image points spreading far away from each other and from the compact set. The latter correspond to the image outliers. Thus, a density based spatial clustering algorithm such as DBSCAN – Density-Based Spatial Clustering of Applications with Noise – (Ester et al., 1996) is applied for removing image outliers. Selection of the DBSCAN than other unsupervised clustering methods is due to its robustness to identify largely spread outliers.

The coordinates of each image point onto the multi-dimensional manifold are estimated through the pair-wise similarity distances obtained by the ORB descriptors. In particular, in our research, the classic multi-dimensional scaling algorithm (Cox and Cox, 2008) is adopted to relate the space of the distances (pair-wise similarity matching) with the space of Gram matrices through which we are able to compute the image coordinates.

Having discriminated image data belonging to the compact subspace against the image outliers, the next step of the proposed content-based filtering algorithm is to partition the compact subspace into regions that

(a) (b) (c)

Figure 7.1: Image retrieval based on images' title and two-step unsupervised clustering (a) Initially retrieved image set from Flickr by using as query the keyword "Porta Nigra", (b) outliers removal using DBSCAN and (c) spectral clustering to discriminate images depicting the rear and the front view of the monument.

contain the most representative geometric perspectives of an object. These representative views are used for a computational efficient 3D reconstruction without spoiling its performance. Spectral clustering is applied in this research to find out the most representative image views of a cultural object. Selection of spectral clustering is due to the fact that it partitions the data so that the maximum coherence among them is achieved (maximizing intra clustering coherence) while simultaneously the minimum coherence among cluster elements (inter cluster) is reached.

7.3.1 *System Architecture*

We propose a new content-based visual filtering algorithm suitable for cultural heritage objects e-documentation and 3D reconstruction. The main research challenge we address is that the reconstruction process is performed using unstructured image data remotely located and distributed over heterogeneous Web platforms. This implies that large portion of image outliers, (objects irrelevant from the cultural asset monument or archaeological site needed to be processed) are encountered in such collections. For instance, in Figure 7.1, we depict some images retrieved from the Flickr multimedia repository as response of the query "Porta Nigra". It is clear that together with the visual data depicting the Porta Nigra monument, several image outliers have been also returned (see Figure 7.1(a)).

Though a 3D reconstruction engine, such as the structure from motion algorithm, can exclude image outliers, the respective computational cost is quadratically increasing with respect to the number of input images. This makes 3D reconstruction process practically impossible to be implemented for real-time application scenarios. To solve this problem, a content-based filtering is required to "sort" the retrieved data according to "their contribution" to the 3D reconstruction. Thus, we first need to discriminate the relevant/irrelevant image data as depicted in Figure 7.1(b) and then, within the relevant image set to localize those images that represent as much as possible the different canonical views (geometric perspectives) of the cultural object as shown in Figure 7.1(c).

Figure 7.2 presents a block diagram of the proposed methodology. The emphasis is given on content-based filtering of cultural heritage objects as clearly depicted with the dotted line framework in Figure 7.2. Initially, a

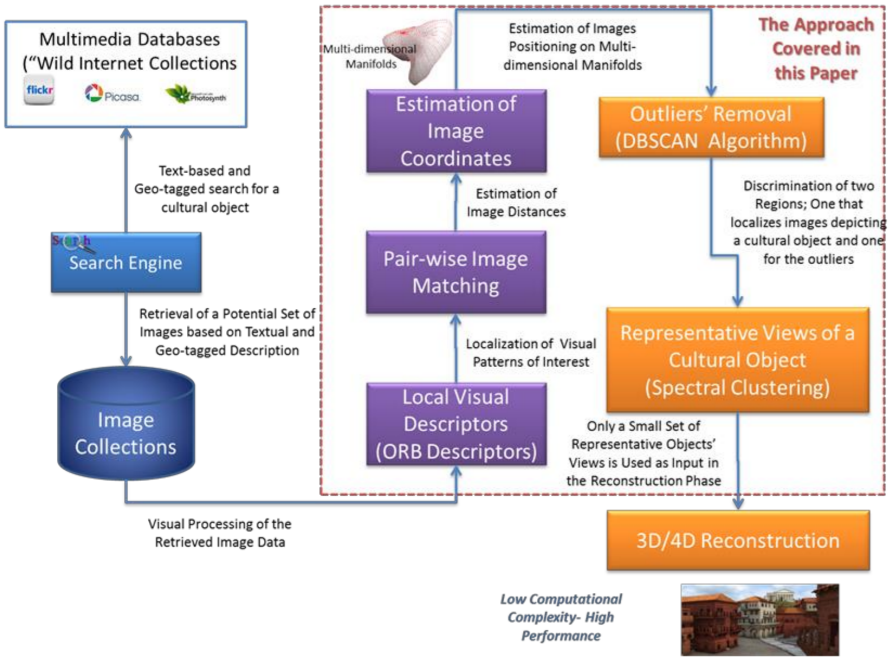


Figure 7.2: The pipeline of the proposed methodology for efficient 3D reconstruction of cultural heritage objects. The dotted framework illustrates the approach being covered by this work.

text-based and geo-tagged based content search algorithm is applied on internet located multimedia databases such as, Flickr, Picasa, Photosynth, etc, upon a user's query for a cultural object of interest. The goal is to retrieve a set of images that can potentially depict the cultural object. As we have stated above, many of these retrieved data are noisily corrupted. This is due to several reasons. For instance, (i) users have generated photos for a monument overlaid with personal content (friends and family being positioning in front of the object), or (ii) they annotate the image content for what they think it depicts (e.g., Parthenon temple is often confused with the Acropolis hill on which it is located), or (iii) they may refer to different salient parts of the site being captured from such location (the Athenian view from the Acropolis hill is often annotated as Acropolis), or (iv) they annotate different objects under the same name (a tavern named Porta Nigra may be retrieved upon submission of such a query to the multimedia databases).

To overcome these problems and to accelerate the reconstruction process, two main objectives should be incorporated. The first removes image outliers while the second discriminates characteristics regions within the relevant image data so that all different geometric views for an object are detected and then fed to the 3D engine. i) Image outliers are excluded using a dense-based clustering algorithm. Conventional clustering algorithms using center-based grouping techniques or spectral analysis, fail to discriminate the outliers from the relevant images since the latter are spread far away from each other and far from the rest of relevant images. ii) The detected compact set of relevant images is then processed to select the M most appropriate elements that maximize 3D reconstruction accuracy. This is performed by selecting as much as "uncorrelated" image data that forms different canonical geometric views of the object. Spectral clustering is adopted to

carry out this task due to its efficiency to define subgraphs with maximum intra cluster and minimum inter-cluster coherency.

7.3.2 *Problem Formulation*

Let us denote as Z a set of images retrieved from distributed located multimedia databases upon a text/geo-tagged based user's query as for a cultural heritage object of interest. As we have stated above, set Z is decomposed into two mutually exclusive sets, $Z = C \cup O$, with $C \cap O = \emptyset$. The set C includes the relevant retrieved images, i.e., those ones whose visual content depicts the object of interest or part of it. Instead, set O stands for image outliers, i.e., images whose content is different than the relevant object. In the proposed automatic content-based filtering algorithm, set Z is known, whereas sets C and O are unknown and should be estimated. Then, our goal is to select, within the relevant set C, a number of M representative images that maximize 3D reconstruction accuracy. It is clear that as number M increases, 3D reconstruction accuracy also increases with, however, a quadratic increase of computational complexity. Thus, for a given number M, we are able to constrain the computational complexity of 3D reconstruction, and then by selecting the M most appropriate images to maximize the respective accuracy.

One way to select the M most representative images is to partition the relevant set C into M mutually exclusive clusters, C_r, $r = 1, 2, ..., M$ each of which is i) as much as "uncorrelated" with samples belonging to different clusters and ii) as much as coherent with samples of the same cluster. This is mainly due to the fact that 3D reconstruction accuracy is maximized in cases that the M selected images present different canonical views of the same object; therefore, the selected M images present no significant redundant information. The aforementioned requirement is mathematically formulated as

$$\hat{C}_r = \min \sum_{i=1}^{M} P_r = \sum_{i \in C_r, j \notin C_r} d_{i,j} \quad \text{and} \quad \max \sum_{i=1}^{M} Q_r = \sum_{i \in C_r, j \in C_r} d_{i,j} \; . \tag{7.1}$$

In Eq.(7.1), $\hat{C}_r$ is the optimal r-th partition of the relevant set C among the M requested, while $d_{i,j}$ is a metric distance between the images $i, j \in C$. The left-hand of Eq.(7.1) minimizes the overall correlation among the M clusters, making their samples as much as "uncorrelated". On the contrary, the right hand of Eq.(7.1) maximizes the coherence within a class.

The main difficulty in solving Eq.(7.1) is that the distances $d_{i,j}$ should be calculated as well as the set C which contains the images i and j. For this reason, we initially need to construct a multi-dimensional manifold on which each image is represented as a point. Then, we exploit the classical Multi-Dimensional Scaling (cMDS) algorithm [(Cox and Cox, 2008) to establish a connection between the distances of image coordinates and the visual matching of different images (see Theorem 1 (Cayton, 2006)). The idea is to relate the visual similarities between two images, as Euclidean distance between two points onto a multidimensional manifold over which the two images are projected.

Then, to estimate set C, we exploit the reasoning that image outliers are spread far away from each other and far from the rest of the relevant images. Thus, center-based clustering algorithms will fail to estimate sets C and O since partitioning on Z will result in many different outliers' clusters

rather than excluding them from the relevant set. For this reason, a density-based clustering algorithm is suitable to perform such clustering, since it is expected that the set C of relevant images will be much denser than the outliers' set O. In this work, a modification of the DBSCAN algorithm (Ester et al., 1996), called Core Sample Partitioning (CSP), is adopted for outliers' detection. The goal of the new approach is to set more strict criteria for partitioning the dataset so that only the most confident image inliers will be included in the relevant dataset. In our case the focus is to create a compact relevant set that contains all different geometric views of an object. Thus, it is more important to exclude all the outliers, which increase computational cost and confuse representative views selection algorithm, from the compact relevant set. On the other hand, considering few relevant objects as outliers cannot affect 3D reconstruction performance if we assume that a sufficient number of relevant images are available.

7.4 GEOMETRIC INVARIANT VISUAL MODELING

Initially, we assume that a set of N images are extracted from the Web, forming the set Z. In our experimentation we have selected multimedia repositories to retrieve image datasets, such as the Flickr, Picasa, etc. These initially N image data are selected using geo-location information as well as textual metadata, meaning that the initially retrieved images share the same textual and geo-location information. Then, a visual based filtering algorithm is applied to select from the set of N initially retrieved images the ones that maximize the performance of the 3D reconstruction engine while keeping as minimum as possible its computational complexity.

This section presents our approach to visually represent the initially retrieved image dataset. In particular, in subsection 7.4.1, we describe the adopted local visual descriptors used to model image content, while in subsection 7.4.2 we formulate the similarity distance between pairs of images. Finally, Subsection 7.4.3 describes the multidimensional scaling algorithm adopted in our research to relate the space distances, expressed through the pair-wise similarity matching, with the space of Gram matrices. The Gram matrices space is used to compute image coordinates onto a multidimensional manifold over which each image is represented.

7.4.1 *ORB-based Visual Content Representation*

Local visual descriptors are used to capture the different geometric perspectives of an object requiring for a 3D reconstruction. The reason of using visual descriptors is to find visual similarity among different images being invariant in affine transformations. This way, we can estimate the distance between two images upon their visual content. We choose to use ORB descriptor (Rublee et al., 2011). Our choice is justified by the fact that, on the one hand, ORB performs better than SURF (Bay et al., 2006) and, on the other, it performs as well as SIFT (Lowe, 2004), while being almost two orders of magnitude faster. ORB descriptor builds on the FAST keypoint detector (Rosten and Drummond, 2006) and the BRIEF descriptor (Calonder et al., 2010) and addresses their limitations: a) it adds a fast and accurate orientation component to FAST, b) it efficiently computes oriented BRIEF features and c) it incorporates a learning method for de-correlating BRIEF features under a rotation invariant framework. For the sake of completeness

and clarity, we briefly describe in the following the ORB descriptor. ORB takes as input an image, it finds image's keypoints, that are image's corners, and then associates each one these keypoints with a description vector.

Let us assume that for each image pixel p_c which have been detected by FAST as corner pixel, a bit-string description is adopted from a set of n binary tests $T = \{\tau_1, \tau_2, ..., \tau_n\}$, where n is a pre-defined scalar parameter of the algorithm. The n binary tests take place in an image patch $l(p_c)$ around the pixel p_c. In particular, we have that $\tau_i(l(p_c); q, r)$ equals one when $I(q) < I(r)$ and zero otherwise. The variables q and r stand for two pixels within the patch $l(p_c)$, while $I(q)$ and $I(r)$ correspond to the image intensities at pixels q and r respectively. Then, a feature is constructed that includes all the n binary tests

$$f_n^{(I)}(l(p_c)) = \sum_{1 \le i \le n} 2^{i-1} \tau_i(l(p_c); q, r) \ . \tag{7.2}$$

By utilizing the intensity centroid corner orientation measure Rosin (1999) the orientation of a patch $l(p)$ around a pixel p is estimated as

$$\theta(l(p_c)) = \arctan(m_{01}(l(p_c)), m_{10}(l(p_c))) \ , \tag{7.3}$$

where $m_{01}(l(p_c))$ and $m_{10}(l(p_c))$ stand for the raw moments of the patch $l(p_c)$.

The projection of the feature vector $f_n^{(I)}(l(p_c))$ onto the angle $\theta(l(p_c))$ results in a rotation invariant binary representation vector, $\boldsymbol{f}_n^{(I)}(l(p_c))$, of patch $l(p_c)$. Then, the visual content of an image I is represented by a matrix

$$\boldsymbol{F}^{(I)} = [\boldsymbol{f}_n^{(I)}(l(p_1)) \ \ \boldsymbol{f}_n^{(I)}(l(p_2)) \ \ \cdots \ \ \boldsymbol{f}_n^{(I)}(l(p_K))]^T \in \{0,1\}^{K \times n} \ . \tag{7.4}$$

7.4.2 *Visual Similarity Degree*

For estimating visual similarity between two different images, A and B, their correspondent points have to be computed. This way, we are able to find images that depict the same cultural heritage object from different geometric perspective point of view. Correspondences can be estimated by performing a nearest-neighbor keypoints matching algorithm between every pair of images. Due to the fact that ORB keypoints are described by a binary pattern, multi-probe Locality Sensitive Hashing (Lv et al., 2007) is used for nearest-neighbor search exploiting the Hamming distance, D_H . Let us denote as $k_i^{(A)}$ the i-th keypoint of the image A (extracted via the ORB algorithm) which is described by the vector $\boldsymbol{f}_n^{(A)}(l(p_i))$. Then, the most relevant keypoint $k_{j_i}^{(B)}$ of the image B with respect to a i-th keypoint of image A, is obtained by the following minimization,

$$j_i = \underset{j=1,...,K}{\operatorname{argmin}} \left(D_H \left(\boldsymbol{f}_n^{(A)}(l(p_i)), \boldsymbol{f}_n^{(B)}(l(p_j)) \right) \right) \ . \tag{7.5}$$

Then keypoints $k_i^{(A)}$ and $k_{j_i}^{(B)}$ are considered as correspondent points.

Having detected all correspondent points between two images A and B we can form a set

$$M^{(A \to B)} = \left\{ \left(k_i^{(A)}, k_{j_i}^{(B)} \right) | i = 1, 2, ..., k \right\} \tag{7.6}$$

that contains all keypoints from the first image A, along with their correspondent keypoints from image B.

For every pair of images in the dataset, a *two-way* matching is performed by following the aforementioned procedure. The set of final matches

$$M^{(A,B)} = M^{(A \to B)} \cap M^{(B \to A)} \tag{7.7}$$

between images A and B is defined to be the intersection of the sets $M^{(A \to B)}$ and $M^{(B \to A)}$.

The choice for using a two-way matching is justified by the fact that the nearest neighbor of an extracted keypoint in image A may be different from the nearest neighbor of the correspondent keypoint in image B. The two way matching compensates such inconsistencies.

Due to the fact that the number of extracted keypoints for each image is equal to K, we define a visual similarity metric between images $i = A$ and $j = B$ as

$$s_{i=A,j=B} = \frac{\left| M^{(A,B)} \right|}{K} \ , \tag{7.8}$$

where $|M^{(A,B)}|$ refers to the cardinality of $M^{(A,B)}$ set.

The output of the aforementioned process for N images is an $N \times N$ symmetric matrix $\boldsymbol{S}$ whose elements $s_{i,j} \in [0, 1]$, $i, j = 1, 2, ..., N$. Variable $s_{i,j}$ takes value of zero in case that the visual content of image i has no relation with the content of image j. Instead, for two similar images variable $s_{i,j}$ takes value equal to one. In the following, we denote by

$$\boldsymbol{D} = [d_{i,j}] = -\log(\boldsymbol{S}) \tag{7.9}$$

the log version of matrix $\boldsymbol{S}$ so as to similar images receive close to zero while quite dissimilar very high value. $\boldsymbol{D}$ is a square $N \times N$ symmetric matrix with non-negative elements and zeros on the main diagonal. In Eq.(7.9), $d_{i,j}$ stands for an element of matrix $\boldsymbol{D}$.

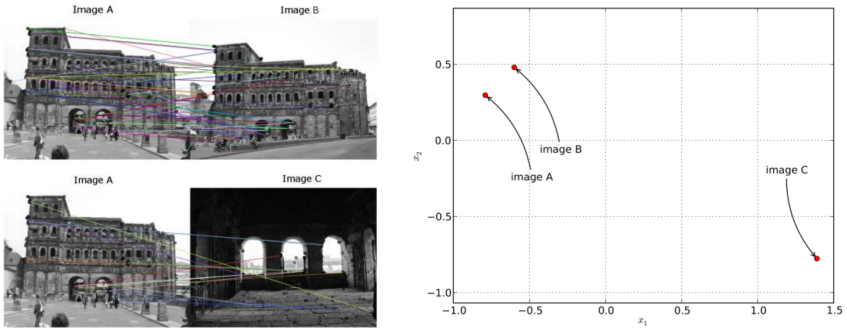
7.4.3 *Image Representation onto Multi-dimensional Manifolds*

By examining the constructed similarity matrix $\boldsymbol{D}$, it is easy to be observed that the distance between the visually similar images is small. This means that if images are represented as points onto a multidimensional manifold, then visually similar images will belong to high spatial density subspaces, instead of image outliers which will be spread out onto the space. Let us define as $\boldsymbol{x}^{(i)} \in \Re^{\mu}$ the coordinates of i-th image in the μ-dimensional space. We define the multi-dimensional space in a way so that the norm (distance) between two points (images) of the space represented by the coordinates $\boldsymbol{x}^{(i)}$ and $\boldsymbol{x}^{(j)}$ should be equal to the their respective image distance $d_{i,j} = -\log(s_{i,j})$ defined by Eq.(7.9), i.e.

$$||\boldsymbol{x}^{(i)} - \boldsymbol{x}^{(j)}|| = d_{i,j}, \quad \forall i, j \ . \tag{7.10}$$

The coordinates of all N images in the dataset can be compactly represented by a matrix

$$\boldsymbol{X} = \begin{bmatrix} (\boldsymbol{x}^{(1)})^T \\ (\boldsymbol{x}^{(2)})^T \\ \vdots \\ (\boldsymbol{x}^{(N)})^T \end{bmatrix} \in \Re^{N \times \mu} \ . \tag{7.11}$$



(a) (b)

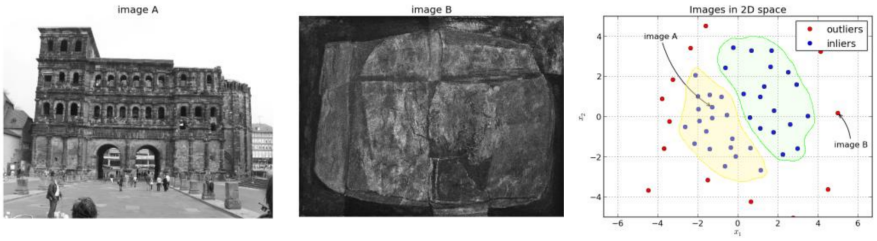
Figure 7.3: (a) ORB descriptors matching for two similar and two dissimilar images. (b) Based on ORB descriptors matching a distance metric between every pair of images is estimated. Using pairwise image distances, images can be represented as points on a multi-dimensional manifold, enabling the exploitation of learning algorithms. For visualization purposes, in this figure, images A, B and C are projected on a 2-dimensional space. Actually our method estimates the number of dimensions directly from the data.

If we define the Gram matrix $\boldsymbol{B} = \boldsymbol{X} \cdot \boldsymbol{X}^T$ of images' coordinates, then classical Multi-Dimensional Scaling (cMDS) (Cox and Cox, 2008) can be used to establish a connection between the space of the distances and the space of Gram matrix $\boldsymbol{B}$ with respect to Eq.(7.10), based on Theorem 1 (Cayton, 2006).

Theorem 1. *A non-negative symmetric matrix $\boldsymbol{D} \in \Re^{N \times N}$, with zeros on the diagonal, is an Euclidean distance matrix if and only if $\boldsymbol{B} := -\frac{1}{2}\boldsymbol{H}\boldsymbol{D}\boldsymbol{H}$, where $\boldsymbol{H} := \boldsymbol{I} - \frac{1}{N}\boldsymbol{1}\boldsymbol{1}^T$, is positive semidefinite. Furthermore, this $\boldsymbol{B}$ will be the Gram matrix for a mean centered configuration with interpoint distances given by $\boldsymbol{D}$.*

In cases where dissimilarity matrix $\boldsymbol{D}$ is not Euclidean the matrix $\boldsymbol{B}$ as described by the above theorem will not be positive semidefinite, and thus will not be a Gram matrix. To handle such cases, cMDS projects the Gram matrix $\boldsymbol{B}$ onto the cone of positive semidefinite matrices by setting its negative eigenvalues to zero. In order to get matrix $\boldsymbol{X}$, the Gram matrix $\boldsymbol{B}$ is spectrally decomposed into $\boldsymbol{U} \cdot \boldsymbol{V} \cdot \boldsymbol{U}^T$ and then $\boldsymbol{X} = \boldsymbol{U} \cdot \boldsymbol{V}^{1/2}$. If we denote as q_i and λ_i for $i = 1, 2, ..., N$ the eigenvectors and eigenvalues of $\boldsymbol{B}$, then matrix $\boldsymbol{U}$ is the square $N \times N$ matrix whose i-th column is the eigenvector q_i of $\boldsymbol{B}$ and $\boldsymbol{V} = [v_{ii}]$ is the diagonal matrix whose diagonal elements v_{ii} are the corresponding eigenvalues, i.e. $v_{ii} = \lambda_i$. Finally, the dimension μ of the multidimensional space is equal to the multiplicity of non-zero eigenvalues of matrix $\boldsymbol{B}$.

Figure 7.3(a) describes an example of visual matching for two images depicting Porta Nigra monument (Image A and B) and two dissimilar images (Image A and C). Based on visual matching, a distance metric between every pair of images is estimated and then, using Theorem 1 these images are projected onto a multidimensional manifold [see Figure 7.3(b)]. (Although Figure 7.3(b) depicts images as points on a 2-dimensional manifold, our method estimates the number of manifold dimensions directly from the data. In Fig. 7.3(b) we used 2-dimensional manifold for visualization purposes). It is clear that the points of the two similar images A and B are located close enough in



(a) (b) (c)

Figure 7.4: Representation of images in a 2-dimensional space. Red dots and blue dots represent image outliers and visually similar images respectively. Points that lie inside the yellow area depict the front view of Porta Nigra monument, while the points that lie inside the green area depict the rear view of the monument. Outliers are scattered and isolated in low density areas, while visually similar images are concentrated in high spatial density areas. For this reason, density partitioning methods are used to isolate image outliers and therefore estimate the compact subset of visually compact images. Instead, conventional clustering methods, such as k-means or spectral clustering fail to remove image outliers since their goal is to partition the high-space into disjoint subsets. For the sake of visualization, in this figure, images are projected on a 2-dimensional space.

the coordinates space, while the points of two dissimilar images A and C far away. Thus, the proposed visual content matching also indicates a "relation degree" among the image points.

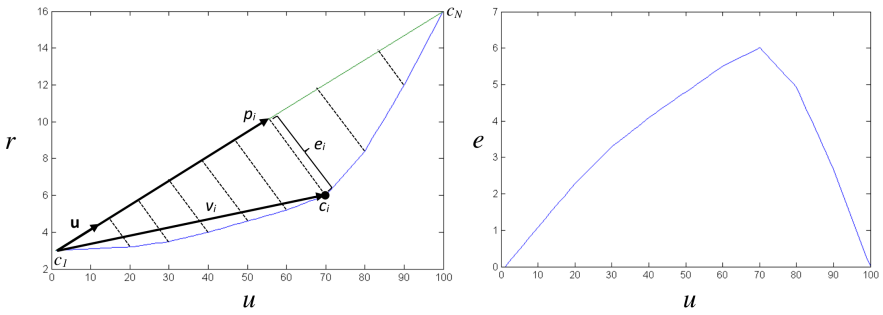
7.5 DENSITY-BASED PARTITIONING FOR EXCLUDING OUTLIERS

By using the representation of images as points onto a multidimensional manifold, we are able to remove image outliers. More specifically, we can intuitively note that outliers reside to areas of low spatial density, due to their large distance from the other images in the dataset. On the contrary, visually similar images form high spatial density areas as depicted in Figure 7.4. Due to this property, partitioning of the multi-dimensional manifold into two disjoint subspaces C (the set of relevant images) and $O = \bar{C}$ (the set of image outliers) cannot be accomplished using conventional center-based clustering techniques (e.g., k-means) or more sophisticated spectral clustering techniques.

For estimating a *compact* subset of images by exploiting *density variations* in space (we focus on the variation of density of the data-points instead of their positioning in the space), partitioning should take place by utilizing a density-based method like the one presented in the following section.

7.5.1 *Estimation of Image Spatial Density*

The density of an area can be defined as the number of points u existed within a specified radius r on the hyperspace manifold. We should mention that r parameter is associated with inter-images distance computed by Eq.(7.9). Estimation of these parameters is crucial for utilizing a density



(a) (b)

Figure 7.5: Estimation of r variable of DBSCAN in regard to u variable. r can be estimated by the best trade-off point of the curve in blue in diagram (a). For finding the best trade-off point the distance of between every curve's point and the straight line defined by the first and the last points of the curve is computed. The point that presents the biggest distance represents the best trade-off point, diagram in (b).

based partitioning algorithm. As no prior knowledge about the dataset is available, these parameters cannot be set to any predefined value. For this reason, we need a procedure to automatically estimate both r and u parameters. In the following, we describe such an algorithm, a visual representation of which is depicted in Figure 7.5.

For a given image A, let us assume that there exists a non-linear relationship, $g^{(A)}(\cdot)$ that relates parameter u with the radius r. Thus, we have that $r = g^{(A)}(u)$. Function $g^{(A)}(\cdot)$ indicates the distance required to be defined for a space in order to contain u points within radius r from image A. This function is plotted in Figure 7.5(a) by the blue line. Function $g^{(A)}(u)$ is monotonically increasing, meaning that as variable u increases the radius r increases too.

Then, to estimate the best trade-off point between u and r of $g^{(A)}(\cdot)$ we adopt the following procedure. First, we define a line segment l that connects the points $\boldsymbol{c}_1 = (u = 1, r = g^{(A)}(1))$ and $\boldsymbol{c}_N = (u = N, r = g^{(A)}(N))$ onto the (u, r) plane (the green line of Figure 7.5(a)). The farthest point of the curve defined by $g^{(A)}(\cdot)$ (blue line of Figure 7.5(a)) from the straight line l (green line Figure 7.5(a)) corresponds to the best trade-off point (Satopaa et al., 2011). To detect this point, initially we define a unit vector as

$$\boldsymbol{u} = \frac{\boldsymbol{c}_N - \boldsymbol{c}_1}{||\boldsymbol{c}_N - \boldsymbol{c}_1||_2} \ . \tag{7.12}$$

It is clear that vector u is parallel to line segment l (see Figure 7.5(a) for clarification). Then, we define as $\boldsymbol{v}_i$ a vector that connects points $\boldsymbol{c}_i = (u = i, r = g^{(A)}(i))$ and $\boldsymbol{c}_1 = (u = 1, r = g^{(A)}(1))$. A geometric clarification of the vector $\boldsymbol{v}_i$ is presented in Figure 7.5(a). The inner product between vector $\boldsymbol{v}_i$ and u, i.e., $\boldsymbol{p}_i = \boldsymbol{u} \cdot \boldsymbol{v}$, projects vector v_i onto the line segment l.

Having estimated the vectors $\boldsymbol{v}_i$ and $\boldsymbol{p}_i$, we are able to compute a distance between vector $\boldsymbol{v}_i$ and its projected version onto the line segment l by

$$e_i = ||\boldsymbol{v}_i - \boldsymbol{p}_i|| \ . \tag{7.13}$$

Figure 7.5(b) plots the distances e_i between curve's points and the straight segment l versus u parameter.

In the following, we denote as $e_{max}^{(A)}$ the maximum value of all e_i that is $e_{max}^{(A)} = \max\{e_1, e_2, ..., e_N\}$ for a given image A. It is clear that for another image a different value of $e_{max}^{(\cdot)}$ is obtained since the non-linear relationship $g^{(\cdot)}(\cdot)$ changes for different images. The aforementioned procedure is repeated for every image of the dataset. Then, the most appropriate image point, say $\hat{i}$ is given by the following equation

$$\hat{i} = \underset{i=1,2,...,N}{\operatorname{argmax}}(e_{max}^{(i)}) \ . \tag{7.14}$$

Then, the most appropriate values of r and u, named $\hat{r}$ and $\hat{u}$ respectively are given by the following equation

$$\hat{u} = \underset{i=1,2,...,N}{\operatorname{argmax}} \left(e_k^{(\hat{i})}\right) \text{ and } \hat{r} = g^{(\hat{i})}(\hat{u}) \ . \tag{7.15}$$

7.5.2 *Core Samples Partitioning*

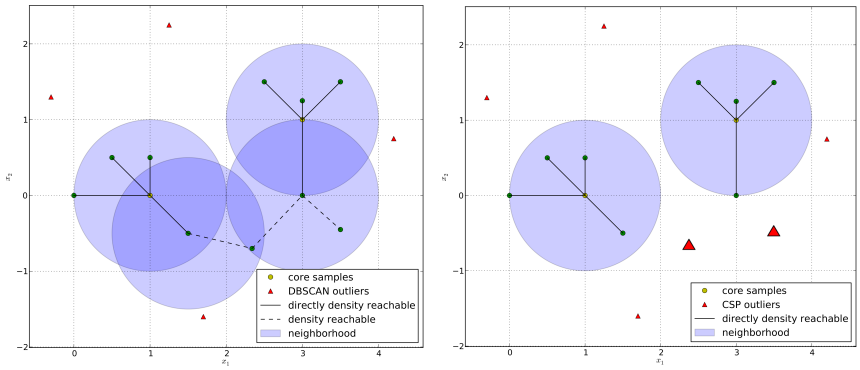
The Core Sample Partitioning (CSP), modification of the DBSCAN algorithm (Ester et al., 1996), is used for outliers' detection. Conventional DBSCAN algorithm partitions the dataset using the spatial density of images onto the multidimensional space. Initially, let us define as $N_r(p)$ a neighborhood of a point $p \in \Re^{\mu}$ (image in our case) on the multidimensional manifold. Then $N_r(p)$ contains all points $q \in \Theta$, whose distance with respect to p is smaller than or equal to radius r,

$$N_r(p) = \{q \in \Theta \mid d_{p,q} < r\} \ , \tag{7.16}$$

where Θ is the set that contains all the N image points onto the multidimensional manifold and $d_{p,q}$ is the distance between two image points p and q, see Eq.(7.10). A point p whose cardinality of neighborhood $|N_r(p)| \geq u$ is called *core* sample. If a point $q \in N_r(p)$ and $|N_r(p)| \geq u$ then points p and q are considered directly *density-reachable*, while two points p and q are considered *density-reachable* if there exists a chain of points $p_1, ..., p_n$ with $p_1 = p$ and $p_n = q$ such that p_i is directly density-reachable from p_{1+1}. Then, the conventional version of DBSCAN creates a compact subset S_{DB} by including all points of the multidimensional manifold that are density reachable from a core sample using the optimized parameters $\hat{r}$ and $\hat{u}$ estimated in Eqs.(7.14)-(7.15). However, creating the subset S_{DB} from points that are density reachable from a core sample, we minimize the probability of an image that contributes to the 3D reconstruction engine (i.e., it depicts a view of the cultural object) to be considered as an outlier. This, however, implies that some of the outliers are included in the target subspace, increasing reconstruction cost.

An alternative approach, named Core Sample Partitioning (CSP), is to set more strict criteria in creating the compact subset S_{DB} of DBSCAN. In particular, CSP exploits the notion of *direct density-reachability*, creating a set say S_{CSP} that minimizes the probability of an image outlier to belong to the partitioned compact subset. In particular, first we define the set C_{cs} that contains all core sample points p_i. Then, based on the set C_{cs} the set S_{CSP} that contains visually similar images is defined as

$$S_{CSP} = \{q \in \Theta \mid d_{p_i,q} < r\} \ , \ \forall p_i \in C_{cs} \ . \tag{7.17}$$



(a) DBSCAN (b) CSP

Figure 7.6: Conventional DBSCAN (a) and CSP partitioning (b). Using DBSCAN all density reachable points from a core sample are considered as inliers. Contrary, by using CSP only the directly density reachable points are denoted as inliers. The large red triangles in (b) correspond to images that were denoted as inliers by using DBSCAN and as outliers by using CSP.

The effect of enabling more strict criteria in creating the compact subset S_{CSP} is shown in Figure 7.6. In this example the parameters r and u have been set to 1 and 4 respectively. Conventional DBSCAN algorithm denotes as inliers twelve points and as outliers four points. When CSP algorithm is applied on the same data, ten points were denoted as inliers and six points as outliers. The two additional outliers that they are detected by CSP and not by DBSCAN are depicted in Figure 7.6(b) as large red triangles. Although, these two points are density reachable from the core samples, they are not directly density reachable and thus they are denoted as outliers using CSP algorithm.

Although CSP partitioning, through the employment of more strict criteria, increases false positive rate during outliers' removal, it minimizes the false negative rate. This means that, on the one hand, it eliminates outliers that will confuse the algorithm which is responsible for selecting the most appropriate views for 3D reconstruction, and on the other it results in a more compact set of visually similar images reducing this way the time required for reconstructing an object. In other words, assuming that a sufficient large set of images depicting a cultural heritage object of interest is available, the proposed modified DBSCAN approach selects images for the 3D reconstruction process that yield low computational complexity, while its precision performance remains almost the same.

7.6 REPRESENTATIVE OBJECT GEOMETRIC PERSPECTIVES

The density based clustering, such as the DBSCAN described in the previous section, discards image outliers through its ability to detect sparse image samples spreading far away from the compact subspace of the "relevant images" onto the multidimensional manifold. Then, we need to partition the detected compact subset of relevant images into different groups that contains the most representative geometric perspectives of the cultural heritage object. This is the goal of this section; to separate image data within this

"relevant region" so that the most representative views of the object, around a sphere that surround it, are determined and then these views are fed as input to a 3D reconstruction algorithm to improve computational complexity while simultaneously keeping the same reconstruction performance. This is presented in Figure 7.4(c).

Spectral clustering is adopted in this approach to achieve such discrimination. The advantages of the spectral clustering algorithm is that it treats clustering as a graph partitioning problem without making specific assumptions on the form of the created clusters. For this reason, let us assume a graph $G = (V, E)$, where V denotes the vertices of the graph, while E the respective edges. In our representation, the vertex set V coincides with the images of the detected compact subset as it has been extracted by the density based clustering algorithm. Thus, the cardinality R of set V equals $R = N - O$, where we recall that N is the total number of images being retrieved based on textual and geo-location criteria, while O is the number of image outliers as they have been detected by the clustering algorithm. We also denote as $w_{i,j}$ the weight of the edge connecting the i-th with the j-th vertex. In our case, the edge weight

$$w_{i,j} = d_{i,j} = -\log(s_{i,j}) \tag{7.18}$$

equals with the similarity distance between the descriptors of the images corresponding to the vertices i and j. Variable $s_{i,j}$ is the similarity matching between the images i and j respectively. Since $w_{i,j} = w_{j,i}$, our graph G is undirected.

Let us recall that the problem we need to solve is given in Eq.(7.1). The main bottleneck of Eq.(7.1) is that creation of small partitions is favored due to the fact that as the number of graph partitions increases, the similarity degree among the partition vertices is expected to increase, while, on the contrary, the similarity among the vertices of different partitions is expected to decrease. To avoid such non-acceptable solution, normalization factors have to be added to Eq.(7.1) (Shi and Malik, 2000). Therefore, considering an M-partitioning problem, we need to estimate clusters that optimize the following quantities

$$\begin{aligned} \hat{C}_r : \max Q &= \sum_{r=1}^{M} NQ_r = \sum_{r=1}^{M} \frac{\sum_{i \in C_r, j \in C_r} d_{i,j}}{\sum_{i \in C_r, j \in C} d_{i,j}} \quad \text{and} \\ \min P &= \sum_{r=1}^{M} NP_r = \sum_{r=1}^{M} \frac{\sum_{i \in C_r, j \notin C_r} d_{i,j}}{\sum_{i \in C_r, j \in C} d_{i,j}} \ , \end{aligned} \tag{7.19}$$

where Q and P is the normalized quantities of Q_r and P_r respectively, while C denotes the union of all partitions C_r, $r = 1, ..., M$, $C = \cup_{r=1}^{M} C_r$.

It can be easily proven, however, that the quantities P and Q are related using the following relationship

$$P + Q = M \ . \tag{7.20}$$

The previous equation implies that maximization of Q simultaneously yields a minimization of P and vice versa. Therefore, it is enough to optimize only one of the two criteria. In the following, we select to minimize the quantity of P.

7.6.1 *Matrix Representation*

In order to minimize Eq.(7.19), we first re-write this equation as a matrix representation form. For this reason, let us denote an index vector $\boldsymbol{a}_r = [\cdots a_r^u \cdots]^T$ whose the u-th entry equals to unity whether the respective u image is assigned to the r-th partition C_r, and zero otherwise;

$$a_r^u = \begin{cases} 1, & \text{if the } u\text{-th image is assigned to } r\text{-th prtition} \\ 0, & \text{otherwise} \end{cases} . \tag{7.21}$$

We also denote as $\boldsymbol{E}$ the adjacent matrix of the graph G. Therefore, we have that the matrix $\boldsymbol{E}$ is expressed as $\boldsymbol{E} = [w_{i,j}]$, where the elements of $w_{i,j}$ are modeled through the Eq.(7.18). Let as also denote as $\boldsymbol{Z}$ the degree matrix of the graph G. The matrix $\boldsymbol{Z}$ is given as a diagonal matrix $\boldsymbol{Z} = diag(\cdots z_i \cdots)$, with

$$z_i = \sum_{j \in C} w_{i,j} . \tag{7.22}$$

Using the degree matrix $\boldsymbol{Z}$, we define the Laplacian matrix of the graph G, $\boldsymbol{L} = \boldsymbol{Z} - \boldsymbol{E}$. Based on the Laplacian matrix, we formulate the numerator of quantity P as a matrix form

$$\boldsymbol{a}_r^T \cdot \boldsymbol{L} \cdot \boldsymbol{a}_r = \sum_{i \in C_r, j \notin C_r} w_{i,j} . \tag{7.23}$$

In the similar way, the denominator of quantity P, (second part of Eq.(7.19)), is formulated as

$$\boldsymbol{a}_r^T \cdot \boldsymbol{Z} \cdot \boldsymbol{a}_r = \sum_{i \in C_r, j \in C} w_{i,j} . \tag{7.24}$$

Therefore, the minimization problem is given in a matrix form as the following equation

$$\hat{\boldsymbol{a}}_r : \min P = \min \sum_{r=1}^{M} \frac{\boldsymbol{a}_r^T \cdot \boldsymbol{L} \cdot \boldsymbol{a}_r}{\boldsymbol{a}_r^T \cdot \boldsymbol{Z} \cdot \boldsymbol{a}_r} . \tag{7.25}$$

7.6.2 *Optimization in the Continuous Domain*

In order to solve the aforementioned optimization problem, we need to relax the index vector of $\boldsymbol{a}_r$ to take continuous values instead of binary ones as presented in Eq.(7.21). This means that we assume that each image is possible to be assigned to all potential clusters but of different degree of membership. Let us denote in the following as $\boldsymbol{I}_R$ the relaxed indicator matrix, $\boldsymbol{I}_R = [\cdots \boldsymbol{a}_r^R \cdots]$ where $\boldsymbol{a}_r^R$ are the relaxed continuous values of $\boldsymbol{a}_r$. The idea is to first find the optimal choice of the relaxed matrix $\boldsymbol{I}_R$, and then discretize somehow the real values to obtain an approximately optimal integer solution. It can be proven (Bach and Jordan, 2003) that the right part of Eq.(7.25) can be rewritten as

$$\begin{aligned} & P = M - \text{trace}(\boldsymbol{Y}^T \cdot \boldsymbol{Z}^{-1/2} \cdot \boldsymbol{E} \cdot \boldsymbol{Z}^{-1/2} \cdot \boldsymbol{Y}) \\ & \text{subject to} \\ & \boldsymbol{Y}^T \boldsymbol{Y} = \boldsymbol{I} , \end{aligned} \tag{7.26}$$

where matrix $\boldsymbol{Y}$ is related to matrix $\boldsymbol{I}_R$ through the equation $\boldsymbol{Z}^{-1/2} \cdot \boldsymbol{Y} = \boldsymbol{I}_R \cdot \boldsymbol{\Lambda}$, where $\boldsymbol{\Lambda}$ is an arbitrary $M \times M$ matrix.

Minimization of Eq.(7.26) is obtained through the Ky-Fan theorem (Fan, 1951), which states that the maximum value of trace$(\boldsymbol{Y}^T \cdot \boldsymbol{Z}^{-1/2} \cdot \boldsymbol{E} \cdot \boldsymbol{Z}^{-1/2} \cdot \boldsymbol{Y})$ with respect to matrix $\boldsymbol{Y}$, subject to the constraint $\boldsymbol{Y}^T \cdot \boldsymbol{Y} = \boldsymbol{I}$ is given by the sum of the M *largest eigenvalues* of matrix $\boldsymbol{Z}^{-1/2} \cdot \boldsymbol{E} \cdot \boldsymbol{Z}^{-1/2}$. Thus, we have that

$$\max\{\text{trace}(\boldsymbol{Y}^T \cdot \boldsymbol{Z}^{-1/2} \cdot \boldsymbol{E} \cdot \boldsymbol{Z}^{-1/2} \cdot \boldsymbol{Y})\} = \sum_{i=1}^{M} \lambda_i \ , \tag{7.27}$$

where λ_i refers to the i-th largest eigenvalue of matrix $\boldsymbol{Z}^{-1/2} \cdot \boldsymbol{E} \cdot \boldsymbol{Z}^{-1/2}$. The Ky-Fan theorem also states that this minimum value is obtained for the matrix

$$\boldsymbol{Y} = \boldsymbol{U} \cdot \boldsymbol{R} \ , \tag{7.28}$$

where $\boldsymbol{U}$ is the matrix whose columns are the eigenvectors corresponding to the M largest eigenvalues of matrix $\boldsymbol{Z}^{-1/2} \cdot \boldsymbol{E} \cdot \boldsymbol{Z}^{-1/2}$ and $\boldsymbol{R}$ is an arbitrary rotation matrix (i.e., orthogonal with determinant of one). Thus, the optimal choice of $\boldsymbol{I}_R$ is (see the aforementioned relations)

$$\hat{\boldsymbol{I}}_R = \boldsymbol{Z}^{-1/2} \cdot \boldsymbol{U} \ . \tag{7.29}$$

In the previous equation, we have assumed that there exists a rotation matrix such that $\boldsymbol{R} \cdot \boldsymbol{\Lambda}^{-1} = \boldsymbol{I}$.

7.6.3 *Solution Discretization*

The optimal matrix $\hat{\boldsymbol{I}}_R$, given by Eq.(7.29), does not have the form of the indicator matrix $\boldsymbol{I}$, since its entries are non-integer, while $\boldsymbol{I}$'s entries are binary. Consequently, the problem is how to round the continuous values of $\hat{\boldsymbol{I}}_R$ in a way that approximates matrix $\boldsymbol{I}$. A simple rounding process is to set the maximum value of each row of $\hat{\boldsymbol{I}}_R$ equal to 1 and the remaining values equal to 0. However, this approach yields unsatisfactory performance when there is no dominant maximum value for each row of $\hat{\boldsymbol{I}}_R$. Furthermore, it handles the rounding process as R independent problems (we recall that R stands for the number of retrieved images excluding the partitions or in other words as the number of rows of $\hat{\boldsymbol{I}}_R$).

An alternative approach, which we adopt in this work, is to treat the R rows of $\hat{\boldsymbol{I}}_R$ as M-dimensional feature vectors. The algorithm clusters the rows of matrix $\hat{\boldsymbol{I}}_R$ to M groups (the number of clusters or the different geometric perspective of the object). Each row of $\hat{\boldsymbol{I}}_R$ indicates the degree of "fitness" (the association degree) of the corresponding image to each of the M clusters. Therefore, the goal of the algorithm is to find the cluster to which an image with a specific feature vector fits best.

It has been shown in (Bach and Jordan, 2003) that such an approach provides the minimum Frobenius distance between the continuous and the discrete solution. Thus, this is the closest approximate solution to the continuous optimum provided by the Ky-Fan theorem. Other methods recursively update the rotation matrix in a way the continuous solution to be as close as possible to the discrete one (Yu and Shi, 2003). However, we avoid applying this method in our case to keep computational complexity as low as possible.

In our case, discretization is achieved via the use of the k-means algorithm on the rows of $\hat{\boldsymbol{I}}_R$. In particular, we initially normalize the rows of matrix $\hat{\boldsymbol{I}}_R$ to take values between 0 and 1. Then, we apply the k-means clustering algorithm to these N vectors to form the indicator matrix $\boldsymbol{I}$. The parameter k of k-means algorithm, i.e. the number of clusters, is set equal to the parameter M. We recall that M is the number of estimated partitions of graph G. Figure 7.4 presents the partitioning of the compact subset into two disjoint sets. In this case the points that lie inside the yellow area correspond to images depicting the front view of the Porta Nigra monument, while the points that lie inside the green area correspond to images depicting monument's rear side. In this case M was set equal to two.

Parameter M is an application defined parameter of the spectral clustering algorithm and it highly affects the accuracy of the following reconstruction method. To be more specific, increment of the value of M leads to the selection of a bigger set of appropriate images for 3D reconstruction, increasing this way the accuracy of the reconstruction algorithm. However, creating a bigger set of representative views for the under reconstruction object, implies that more data are fed to the 3D reconstruction algorithm, increasing this way its computational time requirements.

The estimation of an appropriate value for parameter M takes place in regard to application scenarios. Different scenarios suggest different constraints in regard to devices' computational power and available memory as well as the desired reconstruction time. As mentioned before reconstruction accuracy is monotonically increasing in regard to the value of M. Based on this observation, the parameter M is set to the maximum potential value that satisfies the aforementioned constraints. Consider for example, a tourism application, which is executed on a mobile device. This kind of application imposes strict constraints in regard to computational power and memory requirements as regards 3D reconstruction, resulting in small values of M. The effect of the number of clusters M in the 3D reconstruction accuracy is shown in Figure 7.15. For a given number M, the proposed content based filtering algorithm selects the M most appropriate images to maximize the respective 3D accuracy.

7.6.4 *Selection of the most Representative Images*

The aforementioned algorithm extracts M optimized clusters in a way that each cluster contains images representing a specific geometric perspective of the object of interest. In this way, extracting the most representative images for each cluster, denoted as $I_{r,i}$, $i = 1, 2, ..., M$, we are able to retrieve the most representative geometric views of the under 3D reconstruction cultural heritage object.

In case that we wish to improve the 3D reconstruction accuracy, we should increase the number of retrieved images. One simple way is to re-perform spectral clustering by increasing the number of the extracted partitions M. This, however, implies a high computational cost. An alternative approximate solution is to estimate within each created cluster the most dissimilar image with respect to the most cluster representative. Then, this dissimilar image is selected to be retrieved as the most appropriate.

In particular, let us define as C_i a created cluster using the aforementioned methodology. We recall that as $I_{r,i}$ we denote the most representative image

of this cluster. This image is selected as the one that satisfies the following equation

$$I_{r,i} : \min D(I_{r,i}, \bar{I}_i) = \min d_{I_{r,i},\bar{I}_i} \ , \tag{7.30}$$

where function distance $D(I_{r,i}, \bar{I}_i) = d_{I_{r,i},\bar{I}_i}$ resembles the distance given through Eq.(7.9) and $\bar{I}_i$ is the average image of the cluster C_i produced as the averaging of all images within the cluster. Then, the most dissimilar image is the one that presents the highest distance with the representative image of the cluster;

$$I^d_{r,i} : \max_{I_i \in C_i} D(I_i, I_{r,i}) = \max_{I_i \in C_i} d_{I_i,I_{r,i}} \ . \tag{7.31}$$

We select as the next retrieved image from cluster C_i the one that yields the maximum value of dissimilarity from the previously selected image $I^d_{r,i}$.

7.7 EXPERIMENTAL RESULTS

7.7.1 *Evaluation Metrics and Image Dataset Description*

For the evaluation of the proposed algorithm we use the objective criteria of *Precision*, *Recall* and *F1 Score*. If we denote as C_{vs} the set of ground truth visually similar images and as C_{ret} the set of images that our algorithm denotes as similar then Precision p_r , Recall r_e and F1 Score f_1 metrics are defined as:

$$p_r = \frac{|C_{vs} \cap C_{ret}|}{|C_{ret}|} \ , \ r_e = \frac{|C_{vs} \cap C_{ret}|}{|C_{vs}|} \ , \ f_1 = 2\frac{p_r \cdot r_e}{p_r + r_e} \ . \tag{7.32}$$

In the framework of this research, we have collected from Internet image repositories, (i.e., from "wild image collections"),like Flickr, Picasa, 31,000 images depicted different cultural heritage monuments, archaeological sites, historic places and churches, such as *Porta Nigra* in Germany, *Parthenon* in Athens and *Descobrimentos* in Lisbon, etc. All these images have been gathered with respect to their textual annotation and geo-information regardless of the actual type of content they depict. Thus, for each cultural heritage object category, a large number of image outliers (noise) are encountered (see characteristics examples in Figure 7.1). For example, in the case of Parthenon, noise consists of images that depict anything except the monument.

Using expert's assessment, we have initially annotated this large collection of 31,000 images into two categories; (i) the one of "relevant image set" and (ii) the one of image outliers. The density-based image partitioning algorithm is then applied to discriminate the relevant visual data from the outliers. This way, we create the ground truth set C_{vs} , while the set C_{ret} is constructed by performing the proposed density-based partitioning method. We also exploit the above mentioned metrics of precision, recall and F1 score to evaluate the efficiency of our algorithm and compared it against other approaches. For the evaluation, we range the noise, the percentage of image outliers, in the created datasets from 5% to 60%.

Then, within the annotated relevant image set, we have again categorized the visual data with respect to different geometric views they represent. Our goal is to indicate the number of views (front, back, lateral side of

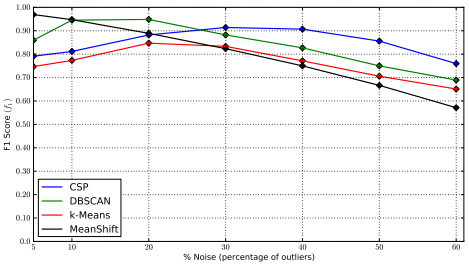


Figure 7.7: F1 Score regarding partitioning performance for outliers' removal using the DBSCAN and CSP along with the k-means and Mean Shift.

an object) required for an acceptable 3D reconstruction process. The constructed ground truth data are used to evaluate the efficiency of the image clustering algorithm applied to estimate the most representative geometric views of an object. This way, we point out a minimum set of representative images needed as input in the 3D reconstruction process so as to achieve maximum reconstruction performance with the minimum possible cost. If we need more images in the reconstruction process the heuristic technique of subsection 7.6.4 can be applied. Then, we evaluate and assess the spectral clustering algorithm for finding out the most representative geometric views of a cultural object/monument as regards cost efficiency and performance. Finally, the impact of the proposed algorithm on 3D reconstruction accuracy and the respective cost needed is shown.

7.7.2 *Evaluation of Partitioning*

For removing the outliers images are placed onto a multidimensional manifold and the spatial density is exploited for partitioning the space into two disjoint subspaces containing the visually similar and non-similar (outliers) images. Obviously, the percentage of outliers in the initial retrieved dataset determines the spatial density of the multidimensional space and thus it affects the performance of the algorithm.

Figure 7.7 presents the F1 Score for two density-based image partitioning approaches for outliers' removal that is, of conventional DBSCAN and its modified method called CSP. The conventional DBSCAN yields better results for small number of outliers (less than 30%) while CSP is more robust in case that the initially retrieved dataset is mostly corrupted with many image outliers. This is due to the fact that the CSP partitioning approach is more prone to false negatives, while DBSCAN is more prone to false positives. The results have been obtained by averaging the values on cultural heritage objects retrieved from Internet collections.

The precision and recall metrics versus the percentage of noise for the two proposed outliers removal methods, i.e., of conventional DBSCAN and of CSP, are shown in Figure 7.8. We observe that precision as regards the CSP approach is higher than the conventional DBSCAN while the opposite is valid as regards the recall metric. These results verify the fact that CSP partitioning is more robust to false positives while less tolerate to false negatives. Again, we average precision, recall values on different cultural heritage objects retrieved from Internet collections in the wild.

Figure 7.9 confirms the aforementioned analysis. In this figure, we depict the percentage of reduction of the initially retrieved image dataset on the use of only textual descriptions and geo-information being available in Internet image databases. Again, the results are figured out against the two proposed outliers' removal techniques, i.e., of conventional DBSCAN and its modified

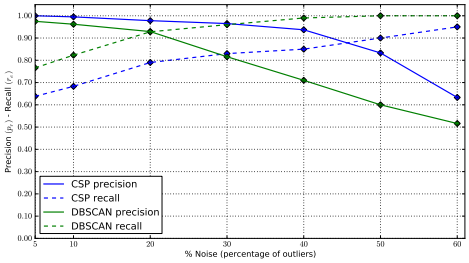


Figure 7.8: Precision and recall diagrams for the two different proposed approaches used for removing outliers.

version of CSP. Conventional DBSCAN results to smaller reduction of the initial retrieved dataset. This is justified by the fact that this approach exploits the notion of density connectivity to form the clusters. Conceptually, partitioning based on density connectivity between images minimizes the number of false negatives but is more prone to false positives and thus it results to larger sets of visually similar images. In addition the percentage of reduction when DBSCAN is used, presents high variations in regard to the percentage of noise, in contrast to the reduction caused by using the CSP method which is less varying with respect to noise fluctuation.

Both conventional DBSCAN partitioning and CSP are compared to k-means and Mean Shift algorithms in Figure 7.7, in terms of *F1 Score*. The k-means is a center-based clustering algorithm, while Mean Shift is a density based one. In the k-means case, we need to define the number of clusters. In this research, we set this value equal to the value that maximizes the silhouette coefficients for different partitions. That is, if we define as a the mean distance between a sample and all other points in the same cluster and as b the mean distance between a sample and all other points in the next nearest cluster, then silhouette coefficient s_c is defined as

$$s_c = \frac{b - a}{\max\{a, b\}} \tag{7.33}$$

Variable a measures how dissimilar is a sample to its own cluster. That is, small values for a means that the respective sample is well matched. Furthermore, a large value for b implies that the sample is badly matched to its neighboring cluster. Thus a silhouette value s_c close to one means that the respective datum is appropriately clustered. Instead, if s_c is close to negative one, then we conclude that the respective sample would be more appropriate to be assigned to its neighboring cluster. A silhouette coefficient value close to zero implies that the sample is on the borders of two clusters.

As shown in Figure 7.7 both conventional DBSCAN and CSP approaches outperform, in terms of F1 score, k-means algorithm. This comparison is aligned with our initial argument that center-based algorithms are inappropriate for identifying outliers. On the other hand, Mean Shift (a density-based algorithm) behaves better, especially for small values of noise (which is not the usual case for Internet based retrieved image collections), while as the noise increases even for intermediate values its performance is severely deteriorates. This is due to the probabilistic nature of Mean Shift which makes it arduous to capture largely spread samples such as the ones being highly corrupted. Hence, when the ratio of noise increases the performance of Mean Shift collapses.

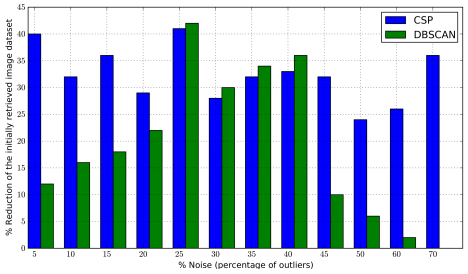


Figure 7.9: Percentage of initial set reduction after the application of outliers' removal approaches, conventional DBSCAN and CSP.

7.7.3 *Evaluation of Image Clustering*

For image clustering, the spectral clustering algorithm is applied on the relevant dataset, i.e., on the dataset obtained having removed the outliers. To evaluate the algorithm performance, we exploit visual data categorization as being obtained with respect to different geometric views. This way, the goal of the algorithm is to extract a limited but characteristic number of images for the cultural object of interest so as to maximize image reconstruction quality, while simultaneously keeping computational cost as low as possible. Figure 7.10 presents spectral clustering performance being measured on the use of precision, recall and *F1 Score* versus noise percentage.

The noise is corresponded to the percentage of image outliers in the initially retrieved image dataset, i.e., on the use of textual descriptions and geo-information from Internet collections. However, the spectral clustering and the precision, recall values are computed on the relevant, reduced set C, that is, the one obtained after excluding the outliers. This is done because noise images, i.e. image outliers, are denoted as such based exclusively on their visual content. Therefore, the ratio of noise in the initial set affects the performance of all components of the system, including local visual descriptors matching, images pairwise distance estimation and thus images representation in a multidimensional space, outliers removal and, finally, spectral clustering, since as noise ratio increases, more outliers are presented in the relevant image set affecting the selection of the most representative views for the under reconstruction object.

This is verified in Figure 7.10 where we observe that precision, recall and *F1 score* achieve their ideal values for noise less than 40%. Instead, if the initially retrieved data are mostly corrupted there is a deterioration of the metrics being however kept on relatively high values for such high corrupted initially retrieved data. In addition, the most important feature of the spectral clustering algorithm is its ability to identify subclasses in the relevant dataset that depict different geometric views of the same object. This way, we can minimize the computational cost needed for a 3D reconstruction while simultaneously keeping the performance as high as possible. Again, the results have been obtained by averaging on cultural heritage objects of our "wild" image dataset.

Figure 7.11, Figure 7.12 and Figure 7.13 visually depict the clustering results of the spectral approach for three monuments, Porta Nigra, Parthenon and Descobrimentos. In all cases, in order to estimate the number of subclasses, we employ the number of connected components in the graph represented by the Laplacian matrix. This estimation takes place by using the eigengap criterion, in order to set M parameter in a fully automatic way, as there is no application specific scenario. This criterion is based on the property of Laplacian matrix, according to which the multiplicity λ of its zero

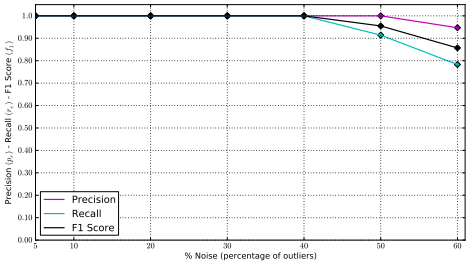
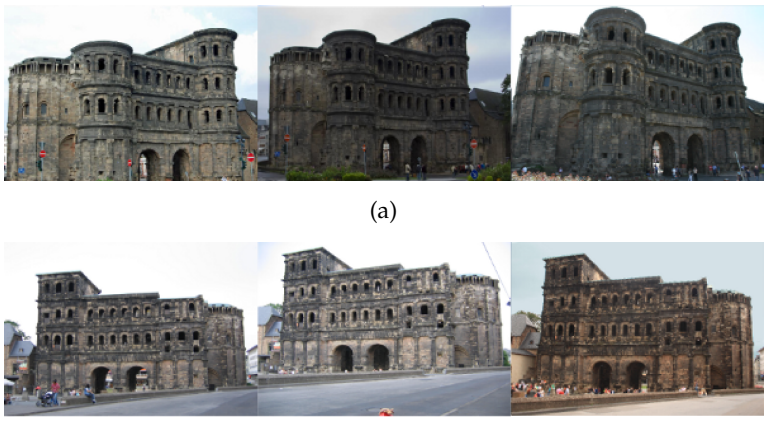


Figure 7.10: Precision, recall and F1 score diagram for spectral clustering algorithm versus the noise of the initially retrieved image collections.



(a)

(b)

Figure 7.11: Clustering results for Porta Nigra monument. The set of relevant images partitioned by using spectral clustering into two disjoint subsets. The first subset (a) includes images that depict the rear side of the monument, while the second subset (b) includes images that depict the front side.

eigenvalues equals to the number of connected components in the graph. Then, we set the number of subclasses equal to the number of connected components in the graph.

Following the aforementioned approach, the relevant images sets for Porta Nigra, Figure 7.11, and Descobrimentos, Figure 7.13, were partition into two disjoint subsets. In the case of Porta Nigra the first subset includes images that depict its front side, Figure 7.11(b), while the second includes images that depict its rear side, Figure 7.11(a). Similar, in the case of Descobrimentos the two disjoint sets contain images that depict its left, Figure 7.13(a), and rear side, Figure 7.13(b). In the case of Parthenon, the relevant images dataset was partitioned into three disjoint sets, each one containing images that depict its front, Figure 7.12(a), rear, Figure 7.12(b) and left-rear, Figure 7.12(c) views. These partitions occurred due to the fact that the vast majority of images, that are not outliers in the initially retrieved dataset, depict these views of the corresponding monuments. Figure 7.13(c) presents an example of an image that was retrieved from the subset of images depicting the rear side of Descobrimentos monument. This image actually shows the same rear view of the monument but, it is as dissimilar as possible to the majority of images belonging to the same subset. For this reason, it can contribute to 3D reconstruction accuracy improvement, by providing the 3D reconstruction method a different perspective of the rear view of the monument.

(a)

(b)

(c)

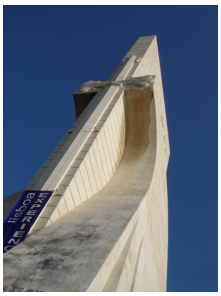
Figure 7.12: Clustering results for Parthenon monument. The set of relevant images partitioned into three disjoint subsets. The first subset (a) includes images that depict the front side of the monument, the second subset (b) includes images that depict its rear side and the third subset (c) includes images that depict the rear and the left side of the monument.

7.7.4 *Impact on 3D Reconstruction Time*

The accuracy of 3D reconstruction in regard to a given image dataset is inherently depended on the number of images that will be fed as input to the Structure from Motion (SfM) scheme. Given an image dataset the best 3D reconstruction accuracy is achieved when all visually similar images are fed into a SfM method. One way to exploit all visually similar images is to give as input to the SfM method the entire image dataset. However, the time complexity for a typical incremental SfM method is of order $O(N^4)$ where we recall that N stands for the number of images. This complexity makes SfM not scalable to large photo collections. In order to decrease computational time required for 3D reconstruction, the initial image dataset can be pruned by removing outliers. When outliers' removal process is very precise (presents recall equal to one), reconstruction accuracy is not affected by dataset reduction, since the relevant (reduced) dataset will contain only all visually similar images. So, the metric that can be associated with reconstruction accuracy is the metric of recall. On the other hand, decrement of SfM computational time is depended on the percentage of reduction of the initial image dataset, which implicitly can be computed by using the metric of precision. When precision metric is equal to one, the cluster of visually similar images contains no outliers, achieving this way the most accurate dataset reduction.

(a)

(b)



(c)

Figure 7.13: Clustering results for Parthenon monument. The set of relevant images partitioned into three disjoint subsets. The first subset (a) includes images that depict the front side of the monument, the second subset (b) includes images that depict its rear side and the third subset (c) includes images that depict the rear and the left side of the monument.

Figure 7.14 presents the recall vs precision diagram when image clustering has been applied on the relevant (reduced) dataset, after removing the outliers utilizing both the conventional DBSCAN and the CSP algorithm. The results have been obtained by averaging on cultural heritage objects of our "wild" image dataset assuming 20% and 30% noise. As this diagram shows both algorithms achieve recall over 90% while at the same time they preserve high precision (almost 90% for DBSCAN and over 95% for CSP). CSP algorithm seems to outperform DBSCAN due to the fact that CSP achieves larger reduction to the initial retrieved image dataset. However, a larger reduction to the initial image dataset may increase the false negative rate of partitioning and consequently affect the accuracy of 3D reconstruction.

Table 7.1 presents a quantitative analysis of the performance of the proposed workflow in terms of precision and recall, percentage of reduction of the initial dataset, reconstruction accuracy and computational time.

We assume that reconstruction accuracy is 100% when all visually similar images of the initial dataset are used. The set of all visually similar images may include redundant information. However, redundant information does not affect 3D reconstruction accuracy. Furthermore, we denote as T the time required for 3D reconstruction when all images of the initial dataset Z are used by a 3D engine. The column titled "Reconstruction accuracy" of Table 7.1 corresponds to the reconstruction accuracy achieved when our approach

is applied on set Z for a given number of M estimated through the eigengap criterion described previously. The column, titled "Computational time on initial set" of the same Table 7.1 corresponds to the time, (also expressed as a portion of T), that our method requires for reconstructing an object, when all images from the whole initial retrieved dataset Z are considered. Finally, the ultimate column, titled "Computational time on ground truth" expresses the time our method requires for reconstructing an object, in comparison with a 3D engine applied on a ground truth dataset (smaller than the initially retrieved image set). Ground truth corresponds to the ideal case, i.e., a set of visually similar images that depict different perspective views of the under reconstruction object.

When the total reduction of the initial dataset is large the computational time for 3D reconstruction is very low. For reduction over 50% the reconstruction process can be 20 times faster. When the reduction of the initial dataset is large (larger than the percentage of noise) the recall metric is low and thus reconstruction accuracy is also low. However, if we constrain recall to be over 90% the system achieves high reconstruction accuracy, while at the same time it achieves more than 4 times faster 3D reconstruction.

For evaluating the representatives selection algorithm, we used expert's assessment in order to select from the set C that contains the visual similar images the n most appropriate for 3D reconstruction: i.e. images correspond to different views of the under reconstruction object. We denote this set as C_n . Then, we asked from our representatives' selection algorithm to extract $n/5$, $2n/5$, $3n/5$, $4n/5$ and n images from the set C. The set of extracted images are denoted as $C_{e,i}$, where $i \in \{n/5, 2n/5, 3n/5, 4n/5, n\}$. The cardinalities of the sets $C_{e,i}$ correspond to different application scenarios. The case where $n/5$ images are selected corresponds to an application scenario that set strict constraints in regard to computational cost, memory requirements and reconstruction time. On the contrary, in the case where n images are selected, corresponds to an application scenario that mainly focuses on reconstruction accuracy, regardless the reconstruction time and/or computational cost.

In this framework reconstruction accuracy is defined as $A = |C_n \cap C_{e,i}|/|C_n|$ where $|\cdot|$ represents the cardinality of a set. By the definition of reconstruction accuracy is obvious that for the cases of $n/5$, $2n/5$, $3n/5$, $4n/5$ and n extracted images, the maximum reconstruction accuracy that can be obtained is 20%, 40%, 60%, 80% and 100% respectively. Furthermore, we compared our representative selection algorithm with k-Means algorithm and spectral clustering using min cut. We request from these algorithms to partition the set C into $n/5$, $2n/5$, $3n/5$, $4n/5$ and n clusters. Then, from each one of the clusters we selected as representative image, the image that is farther from the cluster centroid than the rest images of the same cluster. Evaluation results are shown in Figure 7.15. As the number of clusters is getting larger, the performances of all algorithms are increasing. This is justified by the fact that as the number of clusters is increasing, each one of them contains fewer elements and thus the probability to select the true representative is increasing. Min cut spectral clustering presents the worst performance because it favors unbalanced partitioning. In all cases, our approach based on normalized cut spectral clustering outperforms both k-Means and min cut spectral clustering. Finally, Figure 7.16, presents a 3D reconstruction for the Porta Nigra monument. For this reconstruction 30 images dataset was used that contained 20% of outliers.

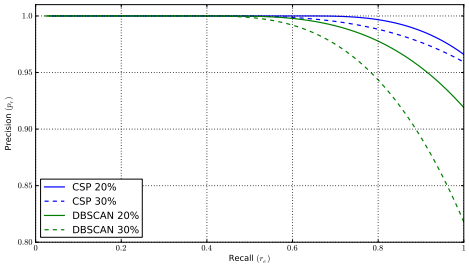


Figure 7.14: Recall vs Precision diagram when initial image set reduction has been performed by using the conventional DBSACN and CSP algorithms.

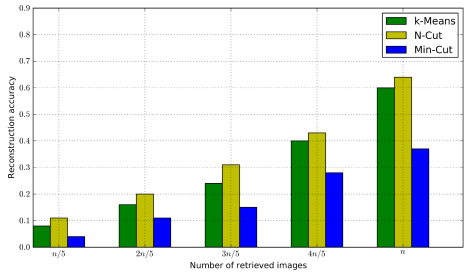


Figure 7.15: :Reconstruction accuracy in regard to the number of selected representatives.

7.8 CONCLUSIONS

The rapid progress in technology regarding visual capturing accompanying with respective progress in respective software tools has stimulated the generation of millions of image content being nowadays stored onto distributed and heterogeneous Internet repositories, like Flickr, Picasa, Photosynth, etc. This content provides a unique opportunity for cultural heritage documentation, like for 3D reconstruction, through the fact that the overwhelming majority of these images have been captured for personal use and thus they are not suitable for such documentation process. Thus, many of these images contain irrelevant material like views of other objects, or of the city instead of the monument itself. This constitutes one of the main innovations of this research, i.e., to allow for reliable and cost effective e-documentation on the use of "wild image collections", that is, data being stored in the Internet under an unstructured way. Therefore, content-based filtering algorithms are necessary for an effective and computationally efficient e-documentation process that exploits the "wild Internet image collections". The main goal of

Noise %	Initial reduction		Precisionon initial set		Recall on initial set		Precision on the Relevant set		Recall on the Relevant set		Total reduction		Reconstruction accuracy		Computational time on initial set		Computational time on ground truth	
	DB	CSP	DB	CSP	DB	CSP	DB	CSP	DB	CSP	DB	CSP	DB	CSP	DB	CSP	DB	CSP
20%	22%	31%	0.92	0.97	0.96	0.81	1	1	0.47	0.65	63.3%	55.2%	45%	52.7%	0.02T	0.04T	0.05	0.09
							0.94	0.97	1	1	22%	31%	96%	81%	0.37T	0.22T	0.89	0.54
							0.94	0.98	0.9	0.9	29%	38%	86.4%	72.6%	0.25T	0.14T	0.61	0.34
30%	30%	28%	0.82	0.96	0.96	0.83	1	1	0.39	0.44	72.7%	68.3%	37.4%	36.5%	0.005T	0.01T	0.02	0.04
							0.82	0.96	1	1	30%	28%	96%	83%	0.24T	0.27T	1.00	1.10
							0.84	0.96	0.9	0.9	38.8%	35.6%	86.4%	74.7%	0.14T	0.17T	0.58	0.71

Table 7.1: Algorithms performance in terms of precision and recall, images' set reduction, reconstruction accuracy and computational time for 3D reconstruction .

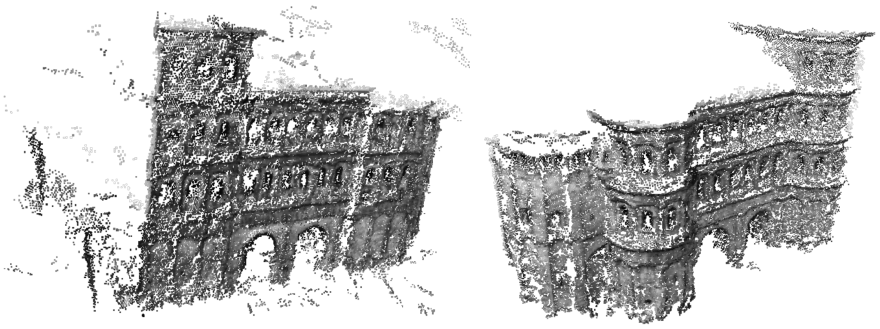


Figure 7.16: 3D reconstruction of rear and front view sides of Porta Nigra. For this reconstruction 30 images were used that contained 20% of outliers.

the proposed content-based image filtering is to discard image outliers that are often retrieved from such Internet collections selecting few but appropriate images needed to be fed as input in the 3D reconstruction process. This way, we minimize computational complexity while keeping performance as high as possible.

Initially, local visual descriptors are extracted to capture different geometric properties and perspectives of an object. In this research, the Oriented FAST and Rotated BRIEF (ORB) descriptor is adopted as a proper combination of the well-know FAST key-point detector and the recently developed BRIEF descriptor, which is rotation invariant and noise resistant. Then, a two-way pair-wise matching is applied resulting in a construction of a similarity matrix. In order to unsupervised remove the image outliers from the retrieved image set (that is, without any knowledge), each image is considered as a point onto a multidimensional hyperspace manifold, the coordinates of which express the position of the images on the hyperspace indicating how close the images are. We estimate the coordinates of these images from their distances calculated via the similarity matrix. In particular, the classical multidimensional scaling algorithm is adopted to relate the space of the image distances with the space of Gram matrices through which we are able to compute the image coordinates. Then, the density-based DBSCAN is applied to remove the outliers and construct a relevant image dataset that includes visual data depicting the object of interest. Then, we partition this "relevant" dataset to find images that contain the most representative geometric perspectives of an object. These representative views are used for a computational efficient 3D reconstruction without spoiling its performance. Spectral clustering is exploited to estimate these representative views.

The system was evaluated by using about 31,000 images mostly retrieved from Flickr, Picasa concerning different cultural heritage objects. Experimental results showed that the system is capable to eliminate outliers from the initial retrieved datasets, even if they contain a large percentage of noise (more than 50%). In addition, the spectral clustering algorithm can define different geometric views of an object reducing the cost of 3D reconstruction without spoiling its performance. We can conclude that for a reduction over 50% the reconstruction process can be 20 times faster.

As for future work, we will investigate new results on accelerating the performance of spectral clustering using incremental learning theory and

Markovian random walk processes. In addition, the effect of different visual descriptors can be further surveyed.

8 ONLINE INDEXING FOR UNSTRUCTURED IMAGE DATA

8.1 MOTIVATION

The volume of the existing image repositories is continuously increasing – users generate new content every day. Thus, the output of a CBIR system that focuses on cultural heritage applications, like the one presented in Chapter 8, after a short time period can be considered obsolete. Furthermore, producing a new updated output requires processing the newly generated data and re-processing the initially retrieved ones. For this reason, it is an imminent need to index image data in a efficient way and provide integrated CBIR and indexing systems capable of dynamically adapting to the continuously evolving nature of image repositories.

We propose an incremental structure scheme able to online index, through the calculation of the visual distance, each new incoming image datum with respect to already indexed image volumes in a fast and accurate way. In this way, we are able to online organize retrieved image data under a computationally efficient manner. The proposed online indexing structure allows for an efficient implementation of meta-algorithms that can incrementally process big and varying image volumes. A content-based filtering approach is presented suitable for selecting appropriate geometric varying images for 3D reconstruction purposes. In particular, our approach exploits the online structure indexing mechanisms to appropriately organize new incoming image data and then adopts geometric properties in a multi-dimensional image manifold (maximize the geometric volume of image points) to select those data that optimize 3D reconstruction operation.

8.2 APPROACH OVERVIEW

The online indexing structure is constructed with the aim to scale large image volumes. For this reason, a pre-defined number of landmark images are selected to represent as much as possible the initially retrieved image data points. We followed the procedure described in Chapter 7 to project images onto a multidimensional manifold and relate their pairwise visual similarity with respect to Euclidean distances. In this multidimensional manifold, image landmarks guarantees that the distance of the newly retrieved image with respect to the remaining indexed ones is able to be computed both computationally efficient under a constant time of operations and effectively.

The position of initially retrieved images on the manifold is a clear indicator of how close the visual content of two images is, see Figure 8.1. The distribution of the retrieved images on the manifold is expected to form i) a compact hyperspace on which images depicting the same object are located and ii) low density areas containing image outliers. In order to develop a robust indexing structure image outliers must be eliminated. This can be done either by following the procedure presented in Chapter 7, or through the exploitation of a density based clustering algorithm. In this work we choose Stochastic Outlier Selection (SOS) Janssens et al. (2012) due to its property to compute the probability that a data point is an outlier. Outlier probabili-

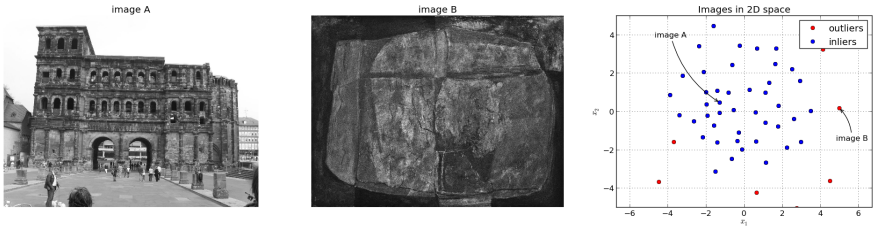


Figure 8.1: Example of two images that were retrieved by using the textual query "Porta Nigra" and their projection on a 2D manifold. Their coordinates were computed by using the distance between them, which was established by local descriptor pair-wise similarity matching. Image A that depicts the monument is positioned in a high density area, while Image B, which is an outlier, is positioned in a low density area.

ties are favorable to unbounded outliers scores and to hard classification of data, because they allow to select an appropriate and rational threshold for outliers selection.

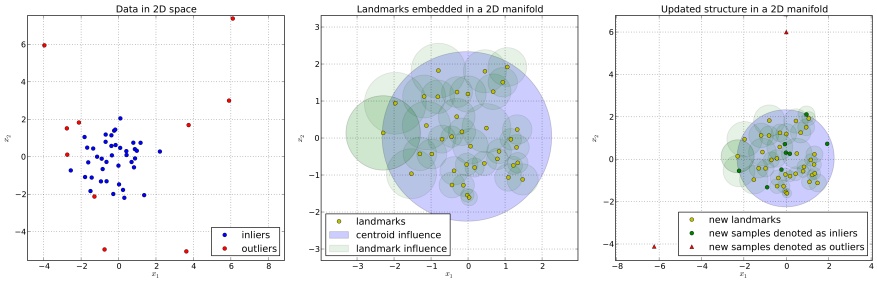
Having discriminated image data to the compact subspace against the image outliers, the next step of the proposed method is to create an appropriate structure capable to index new image samples online. Online indexing aims to a dynamic system capable to scale on large datasets. The indexing structure is constructed by using a predefined number of images, denoted as landmarks, belonging to the compact subspace. Landmarks are used, on the one hand for projecting new image samples on the multidimensional manifold and on the other for denoting new samples as inliers or outliers, see Figure 8.2.

After the indexing of newly retrieved images we must be able to incrementally extract a set of images that are most suitable for 3D reconstruction. A 3D reconstruction engine exploits different geometric perspectives of an object. For this reason, redundant information can be considered as those images presenting similar geometric views of the object to be reconstructed. The incremental set creation enables us to feed the 3D reconstruction engine with the minimum required number of appropriate geometric views of an object so as to achieve a targeted precise reconstruction at a given scale. The selection technique is based on the fact that the volume contained by a simplex formed by the most representative images is larger than any other simplex volume formed by any other combination of images (Winter, 1999).

8.3 THE ONLINE IMAGE INDEXING STRUCTURE

The number of available images stored on Internet multimedia repositories is continuously increasing. For this reason, the proposed method focuses on creating an indexing structure capable to process *online* new retrieved images other than those included in the initial dataset. However, in order to develop a robust indexing structure, we must eliminate image outliers and form a set that will contain only the visually similar images.

By using the representation of images as points onto an μ-dimensional space, we can intuitively note that outliers must reside to low spatial density areas, whereas visually similar images must form areas of high spatial



(a) (b) (c)

Figure 8.2: (a) Images projected in a 2D manifold. Their coordinates were computed by using their pair-wise distances. (b) Inliers were selected as landmarks and defined a new 2D subspace. (c) New samples (red triangles and green circles) are indexed/projected according to landmarks. Green circles correspond to new samples denoted as inliers, while red triangles correspond to new samples denoted as outliers. New samples that fall into the region of influence of centroid or a landmark are denoted as inliers. In the first case the indexing structure remains as it is, while in the second it is updated.

density. Exploiting the density property, or in other words, the affinity between image points, the μ-dimensional manifold must be partitioned into two disjoint subspaces, $\mathcal{C}$ and $\bar{\mathcal{C}}$, such as all visually similar images belong to $\mathcal{C}$ and all outliers to $\bar{\mathcal{C}}$.

8.3.1 *Affinity-based Partitioning*

An affinity-based approach for selecting outliers is the SOS algorithm (Janssens et al., 2012). This algorithm employs the concept of affinity to quantify the relationship from one image point to another. Based on this relationship an image point is denoted as outlier when all other points have insufficient affinity with it.

By using the distance, d_{ij} defined in Eq.(7.9), between image points $\boldsymbol{x}^{(i)}$ and $\boldsymbol{x}^{(j)}$, the affinity between these points can be defined as:

$$\alpha_{ij} = \begin{cases} e^{-(d_{ij}^2/2\sigma_i^2)} & if \;\; i \neq j \\ 0 & if \;\; i = j \end{cases}, \tag{8.1}$$

where σ_i^2 is scalar variance associated with image point $\boldsymbol{x}^{(i)}$. As shown by Eq.(8.1) an image point has no affinity with itself and the affinity that the point $\boldsymbol{x}^{(i)}$ has with point $\boldsymbol{x}^{(j)}$ is proportional to the probability density at $\boldsymbol{x}^{(j)}$ under a Gaussian distribution $\mathcal{N}(\boldsymbol{x}^{(i)}, \sigma_i^2)$. For determining the variance σ_i^2 for each image point, SOS uses an adaptive approach. Concretely, it employs the perplexity parameter h, which is used to set adaptively the variances in such a way that each point has h effective neighbors (Hinton and Roweis, 2002). At this point it has to be mentioned that h is the only parameter that SOS algorithm requires to be pre-defined.

Unlike to distance matrix $\boldsymbol{D}$, the affinity matrix $\boldsymbol{A} = [\alpha_{ij}]$ is not symmetric. By using the affinity distribution $\boldsymbol{\alpha}_i = [\alpha_{i1} \;\; \alpha_{i2} \;\; ... \;\; \alpha_{iN}]$ for the point $\boldsymbol{x}^{(i)}$, a

discrete probability distribution $\boldsymbol{b}_i$ that shows the probability that point $\boldsymbol{x}^{(i)}$ chooses any one of the other points as its neighbors, is defined as

$$\boldsymbol{b}_i = [b_{i1}\ b_{i2}\ ...\ b_{iN}] \quad \text{where} \quad b_{ij} = \frac{\alpha_{ij}}{\sum_{k=1}^{N} \alpha_{ik}}. \tag{8.2}$$

The probability distribution $\boldsymbol{b}_i$ corresponds to the *normalized* affinity $\boldsymbol{\alpha}_i$.

After the estimation of probability distribution $\boldsymbol{b}_i$ the probability the image point $\boldsymbol{x}^{(i)}$ to be denoted as outlier can be estimated by the following theorem (the proof can be found in (Janssens et al., 2012)).

Theorem 1. *If α_{ij} is the affinity that data point $\boldsymbol{x}^{(i)}$ has with data point $\boldsymbol{x}^{(j)}$ and b_{ij} is the normalized affinity between these two points, then the probability that data point $\boldsymbol{x}^{(i)}$ belongs to the outliers class, $\bar{\mathcal{C}}$, is given by:*

$$p(\boldsymbol{x}^{(i)} \in \bar{\mathcal{C}}) = \prod_{j \neq i} (1 - b_{ji}). \tag{8.3}$$

The above theorem states that the probability that an image point $\boldsymbol{x}^{(i)}$ belongs to the outliers class, $\bar{\mathcal{C}}$, is the probability that this point is never chosen as a neighbor of the other image points.

For N images, the output of SOS algorithm can be compactly represented by a vector $\boldsymbol{\rho} \in \mathbb{R}^N$.

$$\boldsymbol{\rho} = [p(\boldsymbol{x}^{(1)} \in \bar{\mathcal{C}}) \quad ... \quad p(\boldsymbol{x}^{(N)} \in \bar{\mathcal{C}})]^T. \tag{8.4}$$

Using Eq.(8.4) the set $\mathcal{Q}$ that will contain the coordinates of the inlier images can be defined as

$$\mathcal{Q} = \{\boldsymbol{x}^{(i)} \mid \rho_i < \theta\} \quad \text{for} \quad i = 1, 2, ...N. \tag{8.5}$$

In Eq.(8.5) ρ_i stands for the i^{th} element of $\boldsymbol{\rho}$ and θ is a probability threshold to discriminate image outliers than inliers.

8.3.2 *Indexing Structure Initialization*

Let us define the set $\mathcal{L} = \{\boldsymbol{x}^{(i)} \mid \boldsymbol{x}^{(i)} \in \mathcal{Q}\}$, which contains visual similar images' coordinates onto the multi-dimensional space. The image points $\boldsymbol{x}^{(i)} \in \mathcal{L}$ act as landmarks that determine if a new image $\hat{I}$ must be denoted as inlier or outlier. The elements of $\mathcal{L}$ define a space with a centroid, c, whose coordinates are $\boldsymbol{x}^{(c)}$. Regions of influence are defined around the centroid and each one of the landmarks. The region of influence of centroid, R_c, is defined as

$$R_c(\boldsymbol{x}^{(c)}, r_c) = \{\boldsymbol{x} \mid (\boldsymbol{x} - \boldsymbol{x}^{(c)})^T (\boldsymbol{x} - \boldsymbol{x}^{(c)}) \leq r_c\}, \tag{8.6}$$

where $r_c = \max\{\|\boldsymbol{x}^{(c)} - \boldsymbol{x}^{(i)}\|_2 \mid \boldsymbol{x}^{(i)} \in \mathcal{L}\}$. In a similar way is defined the region of influence of a landmark $\boldsymbol{x}^{(i)}$

$$R_i(\boldsymbol{x}^{(i)}, r_i) = \{\boldsymbol{x} \mid (\boldsymbol{x} - \boldsymbol{x}^{(i)})^T (\boldsymbol{x} - \boldsymbol{x}^{(i)}) \leq r_i\}. \tag{8.7}$$

In this case r_i is defined as

$$r_i = \min\{\|\boldsymbol{x}^{(i)} - \boldsymbol{x}^{(j)}\|_2 \mid \boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)} \in \mathcal{L} \text{ and } i \neq j\}. \tag{8.8}$$

Regions of influence are used, as described in the next subsection, for classifying new retrieved images as inliers or outliers.

8.3.3 *Online Image Indexing*

Let us assume that a new image, $\hat{I}$ is retrieved. We define the set $\mathcal{Q}_I$ as:

$$\mathcal{Q}_I = \{I^{(i)} \mid \boldsymbol{x}^{(i)} \in \mathcal{Q}\} \tag{8.9}$$

The distances between $\hat{I}$ and each one of the images $I^{(i)} \in \mathcal{Q}_I$ are computed by the method described in Section 2.

In order to index the new image $\hat{I}$, it has to be projected onto the multidimensional geometric space defined by images belonging to $\mathcal{Q}_I$. Let $\hat{\boldsymbol{x}}^{(\hat{I})}$ be the coordinates of image $\hat{I}$ after its projection onto the multidimensional space. The objective of assigning coordinates to image $\hat{I}$ is to minimize the distance distortion given by the following relation:

$$e(I^{(i)}, \hat{I}) = \mid d(I^{(i)}, \hat{I}) - \|\hat{\boldsymbol{x}}^{(\hat{I})} - \boldsymbol{x}^{(i)}\|_2 \mid \tag{8.10}$$

$d(I^{(i)}, \hat{I})$ is the distance between images $I^{(i)}$ and $\hat{I}$ computed by Eq.(7.9) and $\|\cdot\|_2$ refers to the L^2-norm of a vector. Eq.(8.10) measures distance distortion by the absolute error.

The problem of assigning coordinates to image $\hat{I}$ can be seen as a typical optimization problem where the following objective function is minimized.

$$\underset{\hat{\boldsymbol{x}}^{(\hat{I})}}{\operatorname{argmin}} \sqrt{\sum_{i=1}^{L} e(I^{(i)}, \hat{I})^2} \tag{8.11}$$

For estimating the optimal coordinates $\hat{\boldsymbol{x}}^{(\hat{I})}$ we used simplex downhill method. The time for projecting a new image onto an μ-dimensional space is determined by the simplex downhill method. In general simplex downhill with an objective function g takes $O(mD \times f(g))$ time, where $f(g)$ is the cost to evaluate g, D is the number of dimensions and m the number of iterations. In our case, we have $D = \mu$ and $f(g) = L \cdot \mu$, where L stands for the cardinality of $\mathcal{Q}_I$. The second equation holds because we need to calculate the distances between image $\hat{I}$ and each one of the images $I^{(i)} \in \mathcal{Q}_I$ in an μ-dimensional space. In all, the time complexity for indexing a new image is $O(mL\mu^2)$.

Having defined the regions of influence for the centroid and each one of the landmarks (Subsection 3.2), a new image, $\hat{I}$ with coordinates $\hat{\boldsymbol{x}}^{(\hat{I})}$, is denoted as inlier only if $\hat{\boldsymbol{x}}^{(\hat{I})} \in R_c$ or $\hat{\boldsymbol{x}}^{(\hat{I})} \in R_i$ for some $i = 1, 2, ..., |\mathcal{L}|$, where $|\mathcal{L}|$ stands for the cardinality of set $\mathcal{L}$.

If $\hat{\boldsymbol{x}}^{(\hat{I})} \in R_c$ the $\mathcal{L}$ set remains as it is, while $\mathcal{Q}$ and $\mathcal{Q}_I$ sets are updated according to the following relation:

$$\mathcal{Q} := \mathcal{Q} \cup \hat{\boldsymbol{x}}^{(\hat{I})} \quad \text{and} \quad \mathcal{Q}_I := \mathcal{Q}_I \cup \hat{I} \tag{8.12}$$

If $\hat{\boldsymbol{x}}^{(\hat{I})} \in R_i$ for some $i = 1, 2, ..., |\mathcal{L}|$ and $\hat{\boldsymbol{x}}^{(\hat{I})} \notin R_c$ the sets $\mathcal{Q}$ and $\mathcal{Q}_I$ are updated according Eq.(8.12), but in this case the set $\mathcal{L}$ is also updated as:

$$\mathcal{L} := \mathcal{L} \cup \hat{\boldsymbol{x}}^{(\hat{I})} - \min\{\|\boldsymbol{x}^{(i)} - \boldsymbol{x}^{(c)}\|_2 \mid \boldsymbol{x}^{(i)} \in \mathcal{L}\} \tag{8.13}$$

This adaptation takes place for taking into consideration new images visual content, while at the same time keeping constant the number of landmarks.

8.4 REPRESENTATIVE OBJECT GEOMETRIC PERSPECTIVES

After the creation of $\mathcal{Q}$ and $\mathcal{Q}_I$, we need to select the most representative images corresponding to different geometric perspectives of the cultural heritage object under 3D reconstruction. The representative images are fed as input to a 3D reconstruction algorithm to improve computational time while simultaneously keeping the same reconstruction accuracy.

8.4.1 *Representatives Selection through Simplex Volume Expansion*

We assume that the μ-dimensional volume formed by a simplex with vertices specified by the points of the most representative images should be larger than that formed by any other combination of image points. Let us denote as $\boldsymbol{\nu}^{(i)}$ the i^{th} representative image point, as β the number of representative images required to generate, as $\mathcal{Q}_R = \{\boldsymbol{\nu}^{(1)}, \boldsymbol{\nu}^{(2)}, ..., \boldsymbol{\nu}^{(\beta)}\} \subseteq \mathcal{Q}$ the set that contains the representative images' points and as $\boldsymbol{w}^{(j)}$ the row vector that equals to $\boldsymbol{\nu}^{(j)} - \boldsymbol{\nu}^{(1)}$ for $j = 2, 3, ..., \beta$. Then the volume, $V(\mathcal{Q}_R)$, of the simplex whose vertices are the points $\boldsymbol{\nu}^{(i)}$ for $i = 1, 2, ..., \beta$ can be computed as:

$$V(\mathcal{Q}_R) = \frac{|\det(\boldsymbol{W}\boldsymbol{W}^T)|^{1/2}}{(\beta - 1)!} \tag{8.14}$$

where $\boldsymbol{W}$ is an $(\beta - 1) \times \mu$ matrix whose rows are the row vectors $\boldsymbol{w}^{(j)}$.

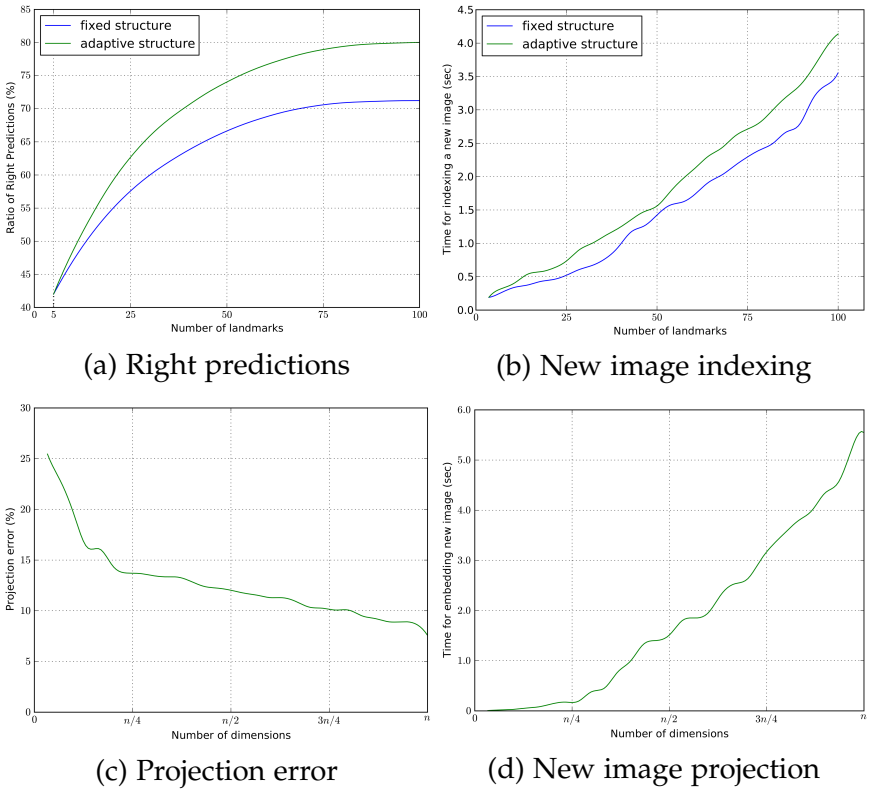
For estimating the most representative images, initially the set $\mathcal{Q}_R$ is constructed by randomly selecting β images from set $\mathcal{Q}$ and calculate the volume of the simplex formed by the elements of $\mathcal{Q}_R$. Then, an iterative approach is adopted to test every image in the set $\mathcal{Q}$ as a candidate representative. To be more specific, each one of the image points of $\mathcal{Q}_R$ is replaced, one at a time, with an image point $\hat{\boldsymbol{\nu}}$ from $\mathcal{Q}$ that is being tested as candidate representative. Then, the algorithm evaluates if replacing any of the elements of $\mathcal{Q}_R$ with the image point being tested results in a larger simplex volume. If this is true, let's say for the point $\boldsymbol{\nu}^{(j)} \in \mathcal{Q}_R$, then the $\boldsymbol{\nu}^{(j)}$ point is replaced by the image point $\hat{\boldsymbol{\nu}}$ and the process is repeated again until each image from $\mathcal{Q}$ set is evaluated.

For making the selection method *scalable* to large datasets, we follow an incremental approach. Let us assume that β representatives are known. Then, the problem of selecting $\beta + 1$ representatives can be reduced to finding $\beta + 1$ representatives *given* β of them. This way, only the volumes of simplices formed by the elements of the sets $\mathcal{Q}_R \cup \boldsymbol{x}^{(i)}$ for $\boldsymbol{x}^{(i)} \in \mathcal{Q}$ need to be evaluated.

8.5 EXPERIMENTAL RESULTS

In the framework fo this research, we have collected from Internet image repositories images depicting different cultural heritage monuments, such as *Porta Nigra* in Germany, *Parthenon* in Athens and *Descobrimentos* in Lisboa. All these images have been gathered with respect to their textual annotation and geo-information regardless of the actual type of content they depict. Thus, for each cultural heritage category, a large number of image outliers are encountered.

The evaluation of the presented approach took place in regard to indexing, in terms of accuracy and time complexity, as well as to 3D reconstruc-



(a) Right predictions

(b) New image indexing

(c) Projection error

(d) New image projection

Figure 8.3: Diagram (a) shows the ratio of right denotations of new images as inliers or outliers in regard to the number of landmarks, while diagram (b) presents the time required to classify a new image. Diagram (c) shows the projection error when assigning coordinates to new images in regard to the number of dimensions of the space onto which the images are projected. The time required to project a new image onto the multi-dimensional space is presented in (d).

tion accuracy after the selection of the most representative images. The algorithm was developed in Python and executed on a conventional i5 CPU laptop.

8.5.1 *Indexing Evaluation*

In order to evaluate indexing mechanism, we created an indexing structure using a varying number of landmarks. Then, we manually selected one hundred outlier images and one hundred inlier images. These images are fed to the indexing mechanism in order to be classified. Two different versions of the algorithm were tested; using a fixed indexing structure and an adaptive indexing structure. In the first case, the indexing structure remains fixed, while in the latter the adaptation mechanism is enabled and the set of landmarks is updated in order to include new images visual information.

Diagram (a) of Fig.(8.3) presents the ratio of right denotations of new images as inliers or outliers in regard to the number of landmarks, while diagram (b) at the same figure shows the time required to classify a new image. The version that uses the adaptive indexing structure is presented

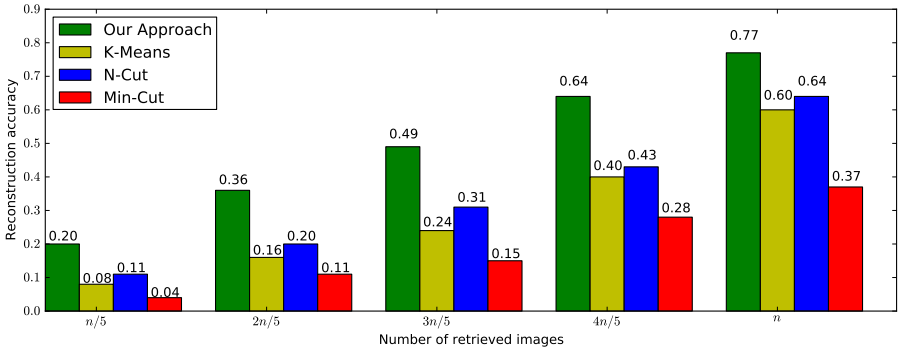


Figure 8.4: This figure presents reconstruction accuracy in regard to the number of selected representatives.

to outperform the one that uses the fixed structure, due to the fact that it exploits visual information of new images. However it requires more time to classify a new image, as it needs extra time to adapt the indexing structure.

Diagrams (c) and (d) of Fig.(8.3) present the projection error when assigning coordinates to new images and the time required to project a new image in regard to the number of dimensions of the space onto which the images are projected. The parameter n in x-axis refers to the number of dimensions of the space. In this case parameter n was set to 100 at the same value was set and the number of landmarks used by indexing structure. As shown in diagram (c) the projection error is constantly decreasing as the number of space dimensions is increasing. In diagram (d) the time required to project a new image onto a multi-dimensional space is increasing as the number of space's dimensions is getting larger. This is aligned with the time complexity analysis presented in subsection 8.3.3.

8.5.2 *Representatives Selection Evaluation*

For evaluating our representatives selection approach, we used expert's assessment in order to select from the set $\mathcal{Q}$ that contain the visual similar images the n most appropriate for 3D reconstruction: i.e. images correspond to different views of the under reconstruction object.

The set of visually similar images contained N elements, and we selected $n = N/5$ of them as the most representatives, set $\mathcal{Q}_r$. Then, we asked from our representatives selection algorithm to extract $n/5$, $2n/5$, $3n/5$, $4n/5$ and n images from the set $\mathcal{Q}$. The set of extracted images are denoted as $\hat{\mathcal{Q}}_i$, where $i \in \{n/5, 2n/5, 3n/5, 4n/5, n\}$ In this framework reconstruction accuracy is defined as $A = |\mathcal{Q}_r \cap \hat{\mathcal{Q}}_i|/|\mathcal{Q}_r|$, where $|\cdot|$ represents the cardinality of a set. By the definition of reconstruction accuracy is obvious that for the cases of $n/5$, $2n/5$, $3n/5$, $4n/5$ and n extracted images, the maximum reconstruction accuracy that can be obtained is 20%, 40%, 60%, 80% and 100% respectively.

Furthermore, we compared our representative selection algorithm with two well known algorithms; K-Means and spectral clustering using normalized cut and min cut. We request from K-Means and spectral clustering algorithms to partition the set $\mathcal{Q}$ into $n/5$, $2n/5$, $3n/5$, $4n/5$ and n clusters. Then, from each one of the clusters we selected as representative image, the

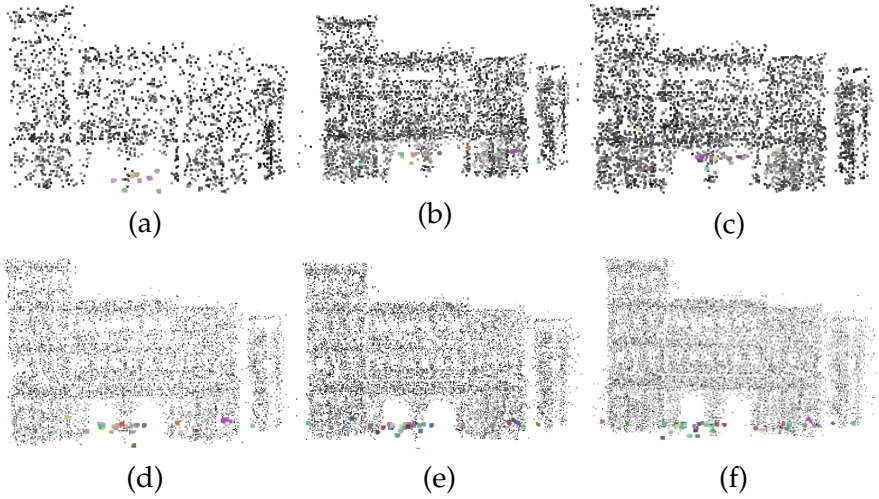


Figure 8.5: (a) - (e) show reconstruction results for *"Porta Nigra"* by selecting $n/5$, $2n/5$, $3n/5$, $4n/5$ and n images using our representatives selection approach. (f) shows reconstruction when all images selected by an expert were used.

image that belongs to $\mathcal{Q}$ and is closer to centroid than the rest images of the same cluster.

Evaluation results are shown in Fig.(8.4). As the number of cluster is getting larger, the performances of K-Means and spectral clustering is increasing. This is justified by the fact that as the number of clusters is increasing, each one of them contains fewer elements and thus the probability to select the true representative is increasing. However, our approach outperforms both algorithms in all cases. Fig.(8.5) shows reconstruction results for *"Porta Nigra"* by selecting $n/5$, $2n/5$, $3n/5$, $4n/5$ and n images using our representatives selection approach.

8.6 CONCLUSIONS

We presented an image indexing approach with application to 3D reconstruction, which is capable to index new images in a *fast* and *accurate* way. Given a set of images, local descriptors are used to encode images' visual content, which, then, is used for estimating a similarity metric between images. This results in the construction of a similarity matrix. Using this similarity matrix images are represented as points into a multi-dimensional space. Exploiting images' coordinates the indexing structure is initialized by eliminating outliers and forming a set of visually similar images. Then, based on the indexing structure, each new retrieved image can be denoted online as inlier or outlier. Furthermore, an accurate algorithm is described for selecting the most appropriate images for 3D reconstruction; i.e. images that depict different views of the same object.

Part III

BEYOND THE VISUAL SPECTRUM

In this part we focus on visual contne analysis using thermal and hyperspectral data. We investigate how information beyond the visual spectrum can be used as a basis for objects detection and event understanding. This investigation takes place in two chapters; the first one presents an algorithm for human detection and tracking, specifically designed to be applied on thermal video streams, while the second one presents a method for material recognition using hyperspectral images.

9 COMPUTER VISION BEYOND THE VISUAL SPECTRUM

9.1 THERMAL AND HYPERSPECTRAL DATA

High level computer vision applications, ranging from video surveillance and monitoring to intelligent vehicles, utilize information that corresponds to visible spectrum. However, under certain environmental conditions, this type of sensing can be severely impaired, which emerges the necessity for imaging beyond the visible spectrum.

Computer vision beyond the visual spectrum focuses on processing data from many different types of sensors, including infrared, far infrared, millimeter wave, microwave, radar and synthetic aperture radar sensors. It involves the creation of new and innovative approaches to the field of signal processing and artificial intelligence. It is a fertile area for growth in both analysis and experimentation. The availability of ever improving computer resources and continuing improvement in sensor performance has given great impetus to this field of research. The dynamics of technology "push" and "pull" in this field of endeavor have resulted from increasing demand from potential users of this technology. Although there are many different sensors capable of capturing information beyond the visual spectrum, in this part we focus on thermal and hyperspectral data processing.

Pixel values of infrared images correspond to the relative differences in the amount of thermal energy emitted or reflected from objects in the scene. Due to this fact, infrared cameras are equally applicable for both day and night scenarios, while at the same time, compared to visual-optical cameras, are less affected by illumination changes. Furthermore, infrared imagery eliminates any privacy issues as people being depicted in the scene can not be identified (Gade et al., 2013). These features, along with the dramatically reduced cost of thermal sensors in the past decades, make infrared cameras prime candidate for persistent video surveillance systems.

Hyperspectral imaging, like other spectral imaging, collects and processes information from across the electromagnetic spectrum. The goal of hyperspectral imaging is to obtain the spectrum for each pixel in the image of a scene, with the purpose of finding objects, identifying materials, or detecting processes based on their spectral signature.

In the following chapters we present the development of a background subtraction algorithm for infrared video sequences, and a deep learning based classification method for hyperspectral data.

10 BACKGROUND SUBTRACTION IN INFRARED IMAGERY

10.1 MOTIVATION

Infrared imagery can alleviate several problems associated with visual-optical videos, however, it has its own unique challenges. Although, less sensitive to lighting conditions than visible spectrum imaging, thermal emission due to the sun illumination as well as the thermal non-homogeneity of objects in the scene add complexity to the visual content (Davis and Sharma, 2004; Pham et al., 2007). Furthermore, the lack of color and texture information hinders low level image processing and impacts on the quality of visual content interpretation (Wang et al., 2010). Finally, low signal-to-noise ratio (noisy data) complicates pixel responses modeling. An example of raw thermal responses is presented in Fig.10.1. Oscillations of pixels' responses witness the presence of noise. Due to these peculiarities most of conventional computer vision techniques, that successfully used for data that correspond to visible spectrum, can not be applied straightforward on infrared imagery.

For many high-level vision based applications, either they use visual-optical videos (Cheung and Kamath, 2005; Porikli, 2006; Tuzel et al., 2007, 2008) or infrared data (Jungling and Arens, 2009; Latecki et al., 2005; Wang et al., 2010), the task of background subtraction constitutes a key component, as this is one of the most common methods for locating moving objects, facilitating, for example, search space reduction and visual attention modeling. Under this scope, the purpose of this work, on the one hand, is to present a novel and high accurate background subtraction algorithm for infrared imagery.

10.2 RELATED WORK

Background subtraction techniques applied on visual-optical videos model the color properties of depicted objects (Brutzer et al., 2011; Herrero and Bescos, 2009) and can be classified into three main categories (El Baf et al., 2009): basic background modeling (McFarlane and Schofield, 1995; Zheng et al., 2006), statistical background modeling (Elgammal et al., 2000; Wren et al., 1997) and background estimation (Messelodi et al., 2005; Toyama et al., 1999). The most used methods are the statistical ones due to their robustness to critical situations. In order to statistically represent the background, a probability distribution is used to model the history of pixel values intensity over time. Towards this direction, the work of Stauffer and Grimson (Stauffer and Grimson, 1999), is one of the best known approaches. It uses a Gaussian Mixture Model (GMM), with fixed number of components, for a per-pixel density estimate. Similar to this approach, Makantasis *et al.* in (Makantasis et al., 2012) propose a Student-t mixture model for background modeling, taking advantage of Student-t distribution compactness and robustness to noise and outliers. The works of (Zivkovic, 2004) and (Zivkovic and van der Heijden, 2006) extend the method of (Stauffer and Grimson, 1999) by introducing a rule based on a user defined threshold to estimate the number of components. However, this rule is application dependent and

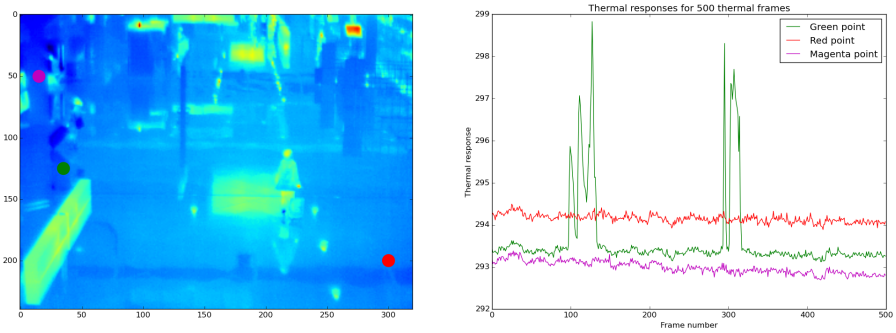


Figure 10.1: Thermal responses for three different points. In contrast to visual-optical videos, where pixels take integer values, thermal responses are floating point numbers, corresponding to objects' temperature.

not directly derived from the data. All of the aforementioned techniques present the drawback that objects' color properties are highly affected by scene illumination, making the same object to look completely different under different lighting or weather conditions.

Although, thermal imagery can provide a challenging alternative for addressing the aforementioned difficulty, there exist few works for thermal data. The authors of (Davis and Sharma, 2005, 2004, 2007) exploit contour saliency to extract foreground objects. Initially, they utilize a unimodal background modeling technique to detect regions of interest and then exploit the halo effect of thermal data for extracting foreground objects. However unimodal background modeling is not usually capable of capturing background dynamics. Baf *et al.* in (El Baf et al., 2009) present a fuzzy statistical method for background subtraction to incorporate uncertainty into the mixture of Gaussians. However, this method requires a predefined number of components making this approach to be application dependent. Elguebaly and Bouguila in (Elguebaly and Bouguila, 2013) propose a finite asymmetric generalized Gaussian mixture model for object detection. Again this method requires a predefined maximum number of components, presenting therefore limitations when this technique is applied on uncontrolled environments. Dai *et al.* in (Dai et al., 2007) propose a method for pedestrian detection and tracking using infrared imagery. This method consists of a background subtraction technique that exploits a two-layer representation (one for foreground and one for background) of infrared frame sequences. However, the assumption made is that the foreground is restricted to moving objects, a consideration which is not sufficient for dynamically changing environments.

One way to handle the aforementioned difficulties is to introduce a background model, the parameters and the structure of which are directly estimated from the data, while at the same time it takes into account the special characteristics of infrared imagery. Furthermore, most of the aforementioned approaches tackle the problem of background subtraction from a theoretical point of view without taking into consideration the computational cost of the algorithm.

10.2.1 *Our Contribution*

This work presents a background modeling method able to provide a per pixel density estimate, taking into account the special characteristics of infrared imagery, such as low signal-to-noise ratio. Our method exploits a Gaussian mixture model with unknown number of components. The advantage of such a model is that its own parameters and structure can be directly estimated from data, allowing dynamic model adaptation to uncontrolled and changing environments.

An important issue using such a model concerns learning its parameters. In our method, this is addressed using a Variational Inference (VI) framework that associates the functional structure of the model with real data distributions obtained from infrared images. Then, the Expectation-Maximization (EM) algorithm is adopted to fit the outcome of VI to real measurements. Updating procedures are incorporated to allow dynamic model adaptation to the forthcoming infrared data. Our updating method avoids of using heuristics by considering existing knowledge accumulated from previous data distributions and then it compensates this knowledge with current measurements.

Our overall strategy exploits a Bayesian framework to estimate all the parameters of the model and thus avoiding over/under fitting issues. To compensate computational challenges arising from the nature of the mixture model (unknown number of components), we utilize conjugate priors and thus we derive analytical equations for model estimation. In this way, we avoid the need of any sampling method, which are computationally and memory inefficient.

10.3 VARIATIONAL INFERENCE FOR GAUSSIAN MIXTURE MODELING

In this section we formulate the Bayesian framework adopted in this work to analytically estimate all the parameters of the background model. For this reason, in section 10.3.1 we briefly describe the basic theory behind Gaussian mixture modeling, while in section 10.3.2 we describe the the variational inference framework that assist us in yielding analytical model estimations as in Section 10.4

10.3.1 *Gaussian Mixture Model Fundamentals*

The Gaussian mixture distribution can be seen as a linear superposition of Gaussian functional components,

$$p(x|\boldsymbol{\varpi}, \boldsymbol{\mu}, \boldsymbol{\tau}) = \sum_{k=1}^{K} \varpi_k \mathcal{N}(x|\mu_k, \tau_k^{-1}), \tag{10.1}$$

where the parameters $\{\varpi_k\}_{k=1}^{K}$ must satisfy $0 \leq \varpi_k \leq 1$ for every k and $\sum_{k=1}^{K} \varpi_k = 1$ and K is the number of Gaussian components. By introducing a K-dimensional latent variable $\boldsymbol{z}$, such as $\sum_{k=1}^{K} z_k = 1$ and $p(z_k = 1) = \varpi_k$, the distribution $p(x)$ can be defined in terms of a marginal distribution $p(\boldsymbol{z})$ and a conditional distribution $p(x|\boldsymbol{z})$ as follows

$$p(x|\boldsymbol{\varpi}, \boldsymbol{\mu}, \boldsymbol{\tau}) = \sum_{\boldsymbol{z}} p(\boldsymbol{z}|\boldsymbol{\varpi}) p(x|\boldsymbol{z}, \boldsymbol{\mu}, \boldsymbol{\tau}), \tag{10.2}$$

where $p(\boldsymbol{z}|\boldsymbol{\varpi})$ and and $p(x|\boldsymbol{z})$ are in the form of

$$p(\boldsymbol{z}|\boldsymbol{\varpi}) = \prod_{k=1}^{K} \omega_k^{z_k}, \tag{10.3}$$

$$p(x|\boldsymbol{z}, \boldsymbol{\mu}, \boldsymbol{\tau}) = \prod_{k=1}^{K} \mathcal{N}(x|\mu_k, \tau_k^{-1})^{z_k}, \tag{10.4}$$

where $\boldsymbol{\mu} = \{\mu_k\}_{k=1}^K$ and $\boldsymbol{\tau} = \{\tau_k\}_{k=1}^K$, correspond to the mean values and precisions of Gaussian components. By introducing latent variables and transforming the Gaussian mixture distribution into the form of (10.2), fitting of the model to the observed data can be achieved through the exploitation of an iterative procedure, such as EM algorithm.

If we have in our disposal a set $\boldsymbol{X} = \{x_1, ..., x_N\}$ of observed data we will also have a set $\boldsymbol{Z} = \{\boldsymbol{z}_1, ..., \boldsymbol{z}_N\}$ of latent variables. Each $\boldsymbol{z}_n$ will be a K-dimensional binary vector, such as $\sum_{k=1}^K z_{nk} = 1$, and, in order to take into consideration the whole dataset of N samples, the distributions of (10.3) and (10.4) will be transformed to

$$p(\boldsymbol{Z}|\boldsymbol{\varpi}) = \prod_{n=1}^{N} \prod_{k=1}^{K} \omega_k^{z_{nk}}, \tag{10.5}$$

$$p(\boldsymbol{X}|\boldsymbol{Z}, \boldsymbol{\mu}, \boldsymbol{\tau}) = \prod_{n=1}^{N} \prod_{k=1}^{K} \mathcal{N}(x_n|\mu_k, \tau_k^{-1})^{z_{nk}}. \tag{10.6}$$

10.3.2 *Distribution Approximation through Variational Inference*

In the framework of VI the model parameters as well as the latent variables are grouped together and are treated as unobserved variables. VI techniques aim to provide an exact analytical solution to an approximation of the posterior probability of unobserved variables given the data (i.e. observed variables) and are utilized when the exact estimation of the posterior is intractable.

Let us denote as $\boldsymbol{Y} = \{\boldsymbol{Z}, \boldsymbol{\varpi}, \boldsymbol{\mu}, \boldsymbol{\tau}\}$ the set which contains all model latent variables and parameters and as $q(\boldsymbol{Y})$ its distribution. The distribution $q(\boldsymbol{Y})$ is restricted to belong to a family of distributions of simpler form than the true posterior $p(\boldsymbol{Y}|\boldsymbol{X})$. The objective is to make $q(\boldsymbol{Y})$ similar to $p(\boldsymbol{Y}|\boldsymbol{X})$. The lack of similarity can be measured by evaluating the quantity

$$KL(q||p) = \int q(\boldsymbol{Y}) \ln \frac{q(\boldsymbol{Y})}{p(\boldsymbol{Y}|\boldsymbol{X})} d\boldsymbol{Y}, \tag{10.7}$$

which corresponds to Kullback-Leibler divergence. $KL(q||p)$ is a non negative quantity, which equals to zero only if $q(\boldsymbol{Y})$ is equal to $p(\boldsymbol{Y}|\boldsymbol{X})$. Thus, attain the aforementioned objective is equivalent to minimizing $KL(q||p)$.

In the framework of the most common type of VI, known as *mean-field variational Bayes*, the variational distribution is assumed to factorize over M disjoint sets such as $q(\boldsymbol{Y}) = \prod_{i=1}^M q_i(\boldsymbol{Y}_i)$. Then, as shown in (Bishop, 2007), the optimal solution $q_j^*(Y_j)$ that corresponds to the minimization of $KL(q||p)$ is given by

$$\ln q_j^*(\boldsymbol{Y}_j) = \mathbb{E}_{i \neq j}[\ln p(\boldsymbol{X}, \boldsymbol{Y})] + \mathcal{C}, \tag{10.8}$$

where $\mathbb{E}_{i \neq j}[\ln p(\boldsymbol{X}, \boldsymbol{Y})]$ is the expectation of the logarithm of the joint distribution over all variables that do not belong to the j^{th} partition and $\mathcal{C}$ is a constant. Equation (10.8) indicates the presence of circular dependencies between the variables that belong to different partitions. Thus, estimating the optimal distribution over all variables suggests the exploitation of an iterative process such as EM algorithm (see Section 10.5).

10.4 OPTIMAL DISTRIBUTIONS OVER MODEL PARAMETERS

In this section, we present the analytical form for the optimal distributions $q_j^*(Y_j)$ for model parameters and latent variables, i.e. the optimal distributions $q^*(\boldsymbol{Z})$, $q^*(\boldsymbol{\varpi})$, $q^*(\boldsymbol{\tau})$ and $q^*(\boldsymbol{\mu}|\boldsymbol{\tau})$, as well as the factorized form of the joint distribution over all random variables.

10.4.1 *Factorized Form of the Joint Distribution*

The joint distribution $p(\boldsymbol{X}, \boldsymbol{Y})$ depends on the distribution form of all model parameters, latent variables and observed data. According to (10.5) and (10.6) the joint distribution can be factorized as follows:

$$p(\boldsymbol{X}, \boldsymbol{Y}) = p(\boldsymbol{X}|\boldsymbol{Z}, \boldsymbol{\mu}, \boldsymbol{\tau}) p(\boldsymbol{Z}|\boldsymbol{\varpi}) p(\boldsymbol{\varpi}) p(\boldsymbol{\mu}, \boldsymbol{\tau}). \tag{10.9}$$

For estimating the form of the factorized distribution, we need to define the form of the prior distribution over $\boldsymbol{\varpi}$ as well as the joint distribution over $\boldsymbol{\mu}$ and $\boldsymbol{\tau}$. To avoid computational problems in estimating the parameters and the structure of our model, we introduce conjugate priors, over $\boldsymbol{\varpi}$, $\boldsymbol{\mu}$ and $\boldsymbol{\tau}$ that allow us to yield analytical solutions and avoid the utilization of computationally expensive sampling methods.

We start our analysis by estimating the form of the prior distribution over $\boldsymbol{\varpi}$. The distribution $p(\boldsymbol{Z}|\boldsymbol{\varpi})$, presented in (10.5), has the form of a Multinomial distribution. Thus, its conjugate prior

$$p(\boldsymbol{\varpi}) = \frac{\Gamma(K\lambda_0)}{\Gamma(\lambda_0)^K} \prod_{k=1}^{K} \varpi_k^{\lambda_0 - 1} \tag{10.10}$$

is a Dirichlet distribution over the mixing coefficients $\boldsymbol{\varpi}$. In (10.10), $\Gamma(\cdot)$ stands for the Gamma function. Furthermore, parameter λ_0 has a physical interpretation; the smaller the value of this parameter is, the larger is the influence of the data rather than the prior on the posterior distribution $p(\boldsymbol{Z}|\boldsymbol{\varpi})$. In order to introduce uninformative priors and not prefer a specific component against the other, we choose to use a single parameter λ_0 for the Dirichlet distribution, instead of a vector with different values for each mixing coefficient.

Similarly, $p(\boldsymbol{\mu}, \boldsymbol{\tau})$ is the prior of $p(\boldsymbol{X}|\boldsymbol{Z}, \boldsymbol{\mu}, \boldsymbol{\tau})$ presented in (10.6). The conjugate prior of (10.6) takes the form of a Gaussian-Gamma distribution, since both $\boldsymbol{\mu}$ and $\boldsymbol{\tau}$ are unknown. Subsequently, the joint distribution of parameters $\boldsymbol{\mu}$ and $\boldsymbol{\tau}$ can be modeled as

$$p(\boldsymbol{\mu}, \boldsymbol{\tau}) = p(\boldsymbol{\mu}|\boldsymbol{\tau}) p(\boldsymbol{\tau}) \tag{10.11a}$$

$$= \prod_{k=1}^{K} \mathcal{N}(\mu_k | m_0, (\beta_0 \tau_k)^{-1}) Gam(\tau_k | a_0, b_0), \tag{10.11b}$$

where $Gam(\cdot)$ denotes the Gamma distribution. In order to not express any specific preference about the form of the Gaussian components through the

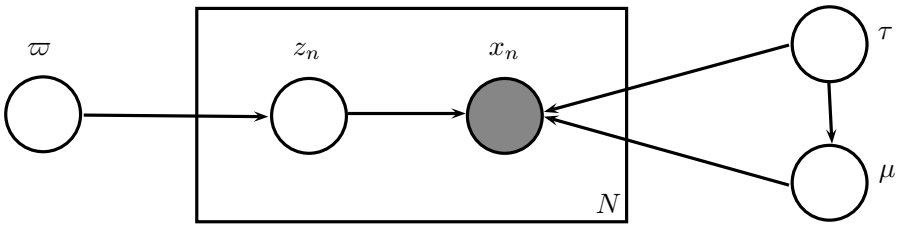


Figure 10.2: Thermal responses for three different points. In contrast to visual-optical videos, where pixels take integer values, thermal responses are floating point numbers, corresponding to objects' temperature.

introduction of priors, we use uninformative priors by setting the values of hyperparameters m_0, β_0, a_0 and b_0 to appropriate values (see Section 10.5).

The graphical probabilistic model that represents (10.9) is presented in Fig.10.2. Directed arrows represent conditional dependencies, the box denotes a set of N independent and identically distributed observations and the shaded circle represents variables that have been set to their observed values.

In the following, the parametric form of (10.9) along with (10.8) are utilized to estimate the optimal variational distributions for model parameters and latent variables [1].

10.4.2 *Optimal $q^*(Z)$ Distribution*

Using (10.8) and the factorized form of (10.9) the distribution of the optimized factor $q^*(\boldsymbol{Z})$ is given by a Multinomial distribution of the form

$$q^*(\boldsymbol{Z}) = \prod_{n=1}^{N}\prod_{k=1}^{K}\left(\frac{\rho_{nk}}{\sum_{j=1}^{K}\rho_{nj}}\right)^{z_{nk}} = \tag{10.12a}$$

$$= \prod_{n=1}^{N}\prod_{k=1}^{K} r_{nk}^{z_{nk}}, \tag{10.12b}$$

where ρ_{nk} we have denote the quantity

$$\rho_{nk} = \exp\Big(\mathbb{E}\big[\ln \varpi_k\big] + \frac{1}{2}\mathbb{E}\big[\ln \tau_k\big] - \frac{1}{2}\ln 2\pi - \\ - \frac{1}{2}\mathbb{E}_{\boldsymbol{\mu},\boldsymbol{\tau}}\big[(x_n-\mu_k)^2\tau_k\big]\Big). \tag{10.13}$$

Due to the fact that $q^*(\boldsymbol{Z})$ is a Multinomial distribution we have that its expected value $\mathbb{E}[z_{nk}]$ will be equal to r_{nk}

10.4.3 *Optimal $q^*(\varpi)$ Distribution*

Using (10.9) and (10.8) the variational distribution of the optimized factor $q^*(\boldsymbol{\varpi})$ is given a Dirichlet distribution of the form

$$q^*(\boldsymbol{\varpi}) = \frac{\Gamma(\sum_{i=1}^{K}\lambda_i)}{\prod_{j=1}^{K}\Gamma(\lambda_j)}\prod_{k=1}^{K}\varpi_k^{\lambda_k-1}. \tag{10.14}$$

1 The derivation of optimal variational distributions over all variables are given in Appendix 10.9.

Variable λ_k is equal to $N_k + \lambda_0$, while $N_k = \sum_{n=1}^{N} r_{nk}$ represents the proportion of data that belong to the k-th component.

10.4.4 *Optimal $q^*(\mu_k|\tau_k)$ distribution*

Similarly, the variational distribution of the optimized factor $q^*(\mu_k, \tau_k)$ is given by a Gaussian distribution of the form

$$q^*(\mu_k|\tau_k) = \mathcal{N}(\mu_k|m_k, (\beta_k\tau)^{-1}), \tag{10.15}$$

where the parameters m_k and β_k are given by

$$\beta_k = \beta_0 + N_k, \tag{10.16a}$$

$$m_k = \frac{1}{\beta_k}\Big(\beta_0 m_0 + N_k \bar{x}_k\Big). \tag{10.16b}$$

Variable $\bar{x}_k$ is equal to $\frac{1}{N_k}\sum_{n=1}^{N} r_{nk}x_n$ and represents the centroid of the data that belong to the k-th component.

10.4.5 *Optimal $q^*(\tau_k)$ distribution*

After the estimation of $q^*(\mu_k|\tau_k)$, the variational distribution of the optimized factor $q^*(\tau_k)$ is given by a Gamma distribution of the following form

$$q^*(\tau_k) = Gam(\tau_k|a_k, b_k), \tag{10.17}$$

while the parameters a_k and b_k are given by the following relations

$$a_k = a_0 + \frac{N_k}{2}, \tag{10.18a}$$

$$b_k = b_0 + \frac{1}{2}\left(N_k\sigma_k + \frac{\beta_0 N_k}{\beta_0 + N_k}(\bar{x}_k - m_0)^2\right), \tag{10.18b}$$

where $\sigma_k = \frac{1}{N_k}\sum_{n=1}^{N}(x_n - \bar{x}_k)^2$.

10.5 DISTRIBUTION PARAMETERS OPTIMIZATION

After the approximation of random variables distributions, we will use the EM algorithm in order to find optimal values for model parameters, i.e. minimize (10.7). In order to use the EM algorithm, we have to initialize priors hyperparameters λ_0, a_0, b_0, m_0 and β_0 and the model parameters ϖ_k, μ_k, τ_k, β_k, a_k, b_k and λ_k (see Section 10.4).

The parameter λ_0 can be interpreted as the effective prior number of observations associated with each component. In order to introduce an uninformative prior for $\boldsymbol{\varpi}$, we set the parameter λ_0 equal to N/K, suggesting that the same number of observations is associated to each component. Parameters a_0 and b_0 (positive values due to Gamma distribution) were set to the value of 10^{-3}. Our choice is justified by the fact that the results of updating equations (10.18a) and (10.18b) are primarily affected by the data and not by the prior when the values for a_0 and b_0 are close to zero. The mean values of the components are described by conditional Normal distribution with means m_0 and precisions $\beta_0\tau_k$. We introduce an uninformative prior by

setting the value for m_0 to the mean of the observed data and the parameter $\beta_0 = \frac{b_0}{a_0 v_0}$, where v_0 is the variance of the observed data.

The convergence of EM algorithm is facilitated by initializing the parameters ϖ_k, μ_k, τ_k and β_k using the k-means. To utilize k-means, the number of clusters, corresponding to the Gaussian components, should be a priori known. Since we create an infinite mixture model, the number of Gaussian components should be less or equal to the number of observed data. For this reason we set the number of clusters K_{max} to a value smaller or equal to the number of observations. If we have no clue about the number of classes we can set K_{max} to equal N. If we denote as $\hat{N}_k$ the number of observation that belong to k-th cluster, then we can set the value of parameter μ_k to equal the centroid of k-th cluster, the parameter ϖ_k to equal the proportion of observations for the k-th cluster, the parameter τ_k to equal $\hat{v}_k^{-1}$, where v_k stands for the variance of the data of the k-th cluster and the parameter β_k to equal $\hat{N}_k^{-1}$. Having initialized the parameters ϖ_k, μ_k, τ_k and β_k, we can exploit the formula for the expected value of a Gamma distribution to initialize the parameters a_k and b_k to values τ_k and one respectively. Finally, the initialization of ϖ_k allows us to initialize the parameter λ_k, which can be interpreted as the effective number of observations associated with each Gaussian component, to the value $N\varpi_k$.

After the initialization of model parameters and priors hyperparameters, the EM algorithm can be used to minimize $KL(q||p)$ presented in (10.7). During the E step, r_{nk} is calculated given the initial/current values of all the parameters of the model. Using (10.12b) r_{nk} is given by

$$r_{nk} \propto \tilde{\varpi}_k \tilde{\tau}_k^{1/2} \exp\left(-\frac{a_k}{2b_k}(x_n - m_k)^2 - \frac{1}{2\beta_k} \right). \tag{10.19}$$

Due to the fact that $q^*(\boldsymbol{\varpi})$ is a Dirichlet distribution and $q^*(\tau_k)$ is a Gamma distribution, $\tilde{\varpi}_k$ and $\tilde{\tau}_k$ will be given by

$$\ln \tilde{\varpi}_k \equiv \mathbb{E}\big[\ln \varpi_k\big] = \Psi(\lambda_k) - \Psi\left(\sum_{k=1}^{K} \lambda_k\right), \tag{10.20a}$$

$$\ln \tilde{\tau}_k \equiv \mathbb{E}\big[\ln \tau_k\big] = \Psi(a_k) - \ln b_k, \tag{10.20b}$$

where $\Psi(\cdot)$ is the digamma function.

During the M step, we keep fixed the value for variables r_{nk} (the value that was calculated during the E step), and we re-calculate the values for model parameters using (10.14), (10.16) and (10.18). The steps E and M are repeated sequentially until the values for model parameters are not changing anymore. As shown in (Boyd and Vandenberghe, 2004) convergence of EM algorithm is guaranteed because bound is convex with respect to each of the factors $q(\boldsymbol{Z})$, $q(\boldsymbol{\varpi})$, $q(\boldsymbol{\mu}|\boldsymbol{\tau})$ and $q(\boldsymbol{\tau})$.

During model training the mixing coefficient for some of the components takes value very close to zero. Components with mixing coefficient less than $1/N$ are removed (we require each component to model at least one observed sample) and thus after training the model has automatically determined the right number of Gaussian components.

10.6 ONLINE UPDATING MECHANISM AND BACKGROUND SUBTRACTION

In the previous sections we described in detail the fitting procedure of the proposed model to N observed data. In this section we present the mecha-

nism that permits our model to automatically adapt to new observed data. The proposed updating mechanism use no heuristic rules, but statistics based exclusively on observations.

10.6.1 *Updating Mechanism using Stored Observed Data*

Let us denote as x_{new} a new observed sample. Then, there are two cases; either the new observed sample is successfully modeled by our fitted model, or not. To estimate if the new sample is successfully modeled, we find the closest component to the new sample. As a distance metric between components and the new sample, we used Mahalanobis distance, since this is a reliable distance measure between a point and a distribution.

The closest component, let us denote it with c, to the new sample is the one that presents the minimum Mahalanobis distance D_k

$$c = \arg\min_k D_k = \arg\min_k \sqrt{(x_{new} - \mu_k)^2 \tau_k}, \tag{10.21}$$

where μ_k and τ_k stand for the mean and precision of k-th component. Then, the new sample belongs to c with probability

$$p(x_{new}|\mu_c, \tau_c) = \mathcal{N}(x_{new}|\mu_c, \tau_c^{-1}), \tag{10.22}$$

where μ_c and τ_c stand for the closest component mean value and precision respectively.

Let us denote as Ω the initially observed dataset, i.e. the pixel responses over a fixed time span. Then, we can assume that the probability to observe the new sample x_{new} is given by

$$p(x_{new}|e) = \frac{N_e}{N} \mathcal{U}(x_{new}|x_{new} - e, x_{new} + e), \tag{10.23}$$

where $N_e = \big|\{x_i \in \Omega : x_{new} - e \leq x_i \leq x_{new} + e\}\big|$ is the cardinality of the set contains samples e-close to x_{new} and $\mathcal{U}(x_{new}|x_{new} - e, x_{new} + e)$ is a Uniform distribution with lower and upper bounds to equal $x_{new} - e$ and $x_{new} + e$ respectively.

Equation (10.23) suggests that the probability to observe x_{new} is related to the proportion of data that have already been observed around x_{new}. By increasing the neighborhood around x_{new}, i.e. increasing the value of e, the quantity $\mathcal{U}(x_{new}|x_{new} - e, x_{new} + e)$ is decreasing, while the value of N_e is increasing.

Upon arrival of a new sample x_{new}, we can estimate the range ϵ around x_{new} that maximizes (10.23)

$$\epsilon = \arg\max_e p(x_{new}|e). \tag{10.24}$$

Then, by comparing $p(x_{new}|\mu_c, \tau_c)$ to $p(x_{new}|\epsilon)$ we can conclude if the new sample can be modeled by our fitted model, or not. Concretely, if $p(x_{new}|\mu_c, \tau_c) \geq p(x_{new}|\epsilon)$ the new observed sample x_{new} can sufficiently represented by our model, or in other words, the value of the new observed sample is sufficiently close to an emerged Gaussian component. Otherwise, a new Gaussian component must be created, since the value of x_{new} will be no close to what the model has already known.

For model updating, either the new sample is modeled or not by the already fitted model, the binary variable o, called ownership and associated with the Gaussian components, is needed to be defined

$$o_k = \begin{cases} 1, & \text{if } k = c \\ 0, & \text{otherwise} \end{cases}, \tag{10.25}$$

where we recall that c represents the index of the closest component and k is the index of k-th component.

When the new observed sample is successfully modeled, the parameters for the Gaussian components are updated using the *following the leader* (Dasgupta and Hsu, 2007) approach described as

$$\varpi_k \leftarrow \varpi_k + \frac{1}{N}(o_k - \varpi_k), \tag{10.26a}$$

$$\mu_k \leftarrow \mu_k + o_k\left(\frac{x_{new} - \mu_k}{\varpi_k N + 1}\right), \tag{10.26b}$$

$$\sigma_k^2 \leftarrow \sigma_k^2 + o_k\left(\frac{\varpi_k N (x_{new} - \mu_k)^2}{(\varpi_k N + 1)^2} - \frac{\sigma_k^2}{\varpi_k N + 1}\right), \tag{10.26c}$$

where σ_k^2 is the variance of k-th components and equals τ_k^{-1}. After the adaptation of existing components mixing coefficients ϖ are normalized to sum to one.

When the new observed sample cannot be modeled by the existing components, a new component is created with mixing coefficient ϖ_{new}, mean value μ_{new} and standard deviation σ_{new}, the parameters of which are given as

$$\varpi_{new} = \frac{1}{N}, \tag{10.27a}$$

$$\mu_{new} = x_{new}, \tag{10.27b}$$

$$\sigma_{new}^2 = \frac{(2\epsilon)^2 - 1}{12}. \tag{10.27c}$$

Variable σ_{new}^2 is being updated using the formula for variance of the Uniform distribution. From (10.27), we see that the mixing coefficient for the new component is equal to $1/N$ since it models only one sample (the new observed one), its mean value equals the value of the new sample and its variance the variance the Uniform distribution, whose the lower and upper bounds are $x_{new} - \epsilon$ and $x_{new} + \epsilon$ respectively. When a new component is created the values for the parameters for all the other components remain unchanged except from the mixing coefficients $\{\varpi_k\}_{k=1}^K$ which are normalized to sum $\frac{N-1}{N}$. Then, the components whose mixing coefficient is less than $\frac{1}{N}$, they model less than one sample, are removed, and the mixing coefficients of the remaining components are re-normalized. This procedure guarantees that after each adaptation of the system to new observed samples, either they modeled by the trained model or not, the sum of the mixing coefficients of the components equals one.

10.6.2 *Updating Mechanism without Keeping Observed Data*

Based on the aforementioned formulation (10.23) of adaptation mechanism, and on the already trained Gaussian mixture model, we can estimate $p(x_{new}|\epsilon)$

without the need of storing observations. This is a crucial step towards implementing our proposed system on low-end hardware devices.

We recall that we have denoted as c the closest component, in terms of Mahalaobis distance, to the new observed datum x_{new}. This component is a Gaussian distribution with mean value μ_c, precision τ_c and mixing coefficient ϖ_c. Therefore, the quantity N_e can be approximated as

$$N_e \approx \tilde{N}_e = N\varpi_c \int_{x_{new}-e}^{x_{new}+e} \mathcal{N}(t|\mu_c, \tau_c^{-1})dt. \tag{10.28}$$

Denoting as

$$G_c(x) = \int_{-\infty}^{x} \mathcal{N}(t|\mu_c, \tau_c^{-1})dt \tag{10.29}$$

the cumulative distribution of the closest Gaussian component, $\tilde{N}_e$ is equal to

$$\tilde{N}_e = N\varpi_c\big(G_c(x_{new}+e) - G_c(x_{new}-e)\big) \tag{10.30}$$

and $p(x_{new}|\epsilon)$ is approximated as

$$\begin{aligned} p(x_{new}|e) &\approx \tilde{p}(x_{new}|e) = \\ &= \frac{\tilde{N}_e}{N}\mathcal{U}(x_{new}|x_{new}-e, x_{new}+e). \end{aligned} \tag{10.31}$$

$\tilde{p}(x_{new}|\epsilon)$ is a continuous and unimodal function with a global maximum. Therefore, ϵ can be found by setting the first derivative of (10.31) equal to zero:

$$\epsilon = \Big|_{\frac{\partial\tilde{p}(x_{new}|e)}{\partial e}=0}. \tag{10.32}$$

After the estimation of ϵ, we can compute $\tilde{p}(x_{new}|\epsilon)$. Then, we are able to update the mixture model by comparing $\tilde{p}(x_{new}|\epsilon)$ to $p(x_{new}|\mu_c, \tau_c)$ and following the same procedure that was described in the previous subsection.

10.6.3 *Background Subtraction*

Let us denote as bg and fg the classes of background and foreground pixels respectively. The density distribution of the fitted mixture model corresponds to the probability $p(x|bg)$. However, our goal is to calculate the probability $p(bg|x)$, hence the Bayes rule is applied;

$$p(bg|x) = \frac{p(x|bg)p(bg)}{p(x|bg) + p(x|fg)}. \tag{10.33}$$

The probability $p(bg)$ corresponds to the prior probability of background class and can be considered as a parameter. The probability $p(x|fg)$ is unknown and hard to estimate, so the uniform distribution over the range of pixels' responses is used (Haines and Xiang, 2014). Using Bayes rule (10.33) the output of the proposed background subtraction algorithm corresponds to the probability a pixel to belong to the background class. The overview of the proposed scheme is shown in Algorithm 1.

Algorithm 1: Overview of Background Subtraction

1: capture N frames
2: create N-length history for each pixel
3: initialize parameters (see Section 10.5)
4: **until** convergence (training phase: Section 10.5)
5: compute r_{nk} using (10.19)
6: recompute parameters using (10.14), (10.16) and (10.18)
7: **for each** new captured frame
8: classify each pixel as foreground or background (see subsection 10.6.3)
9: update background model (see subection 10.6.2)

10.7 EXPERIMENTAL VALIDATION

In this section we evaluate and experimentally validate the proposed model. We evaluate the capability of VI Mixture Model (VIMM) to fit the observed data and compare it to conventional GMM and Dirichlet Process Mixture Model (DPMM). Then, the performance of the proposed updating mechanism and is compared against the GMM updating process presented in (Zivkovic, 2004). We demonstrate the efficiency of the background subtraction process using real-world datasets and, finally, we examine the hardware cost.

10.7.1 *VI Mixture Model Fitting Capabilities*

During experimental validation, we evaluated our model in terms of fitting accuracy and computational cost. We compared our algorithm to conventional GMM with fixed number of components and with DPMM, which fits the data using collapsed Gibbs sampling. We examined the performance of conventional GMM under three different settings, that is, use of GMM equipped with the right number of components, and GMMs with more and less components than the underlying distribution.

For experimentation purposes we created three different datasets, each one contains 200 samples. The first dataset contains samples from two different well separated Gaussian distributions, with mean values 15 and 35, variance 1.5 and 2.0 and proportions 8/15 and 7/15 of data respectively. The second dataset contains samples from three different well separated Gaussian distributions, with mean values 15, 35 and 55, variance 1.5, 2.0 and 2.5 and proportions 4/15, 4/15 and 7/15 of data respectively. Finally, the third dataset contains samples from three different Gaussian distributions, but two of them are not well separated, with mean values 15, 21 and 40, variance 1.5, 1.5 and 1.5 and proportions 7/15, 2/15 and 6/15 of data respectively.Regarding data fitting, the results are shown in Figures 10.3, 10.4 and 10.5.

In all cases the initial value for the number of components for our method was set to the value of 10, and let the fitting procedure to estimate the appropriate number of components that characterize the underlying distribution. In order to compare our method to conventional GMMs with fixed number of components we created one GMM with 10 components and one with 2.

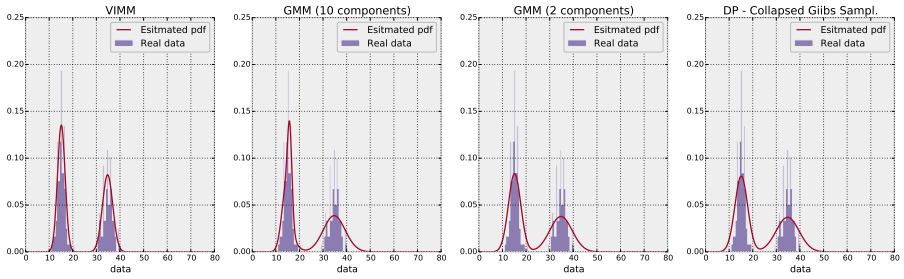


Figure 10.3: Fitting performance - two well separated Gaussian distributions.

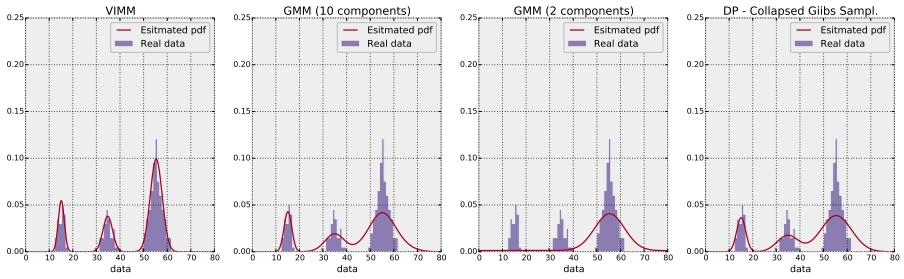


Figure 10.4: Fitting performance - three well separated Gaussian distributions.

Figure 10.3 presents the fitting performance for all four models on the first dataset. Our method correctly estimates the number of components, the same happens for the DPMM. However, the DPMM seems to under-fit the data. The GMM with 10 components fits well the data that come from the first distribution, but under-fits the data that come from the second. Furthermore, the GMM with 10 components seems to estimate two Gaussian distributions, but actually uses 10 components with 30 parameters; mean value, standard deviation and proportion of data for each component. The GMM with 2 components fits well the data. However, the number of components was known *a priori*.

Figure 10.4 presents the fitting performance for all four models on the second dataset. In this dataset the two conventional GMMs with fixed number of components under-fit the data. The same happens for the DPMM, although ti estimates correctly the number of components. On the contrary to the aforementioned models, our model correctly estimates the number of components that describe the underlying distribution and neither underfits nor overfits the data.

Figure 10.5 presents the fitting performance for all four models on the third dataset. The GMM that uses 2 components under-fits the data,while the GMM that uses 10 components highly over-fits the data. Our method fails to discriminate overlapping distributions and results to 2 components. Finally, the DPMM correctly estimates the number of components and fits the data better than all other methods.

Finally, Table 10.1 presents time performance of the different models. It has to be mentioned that all presented times were computed in Python and not in hardware implementation. The conventional GMMs present the best

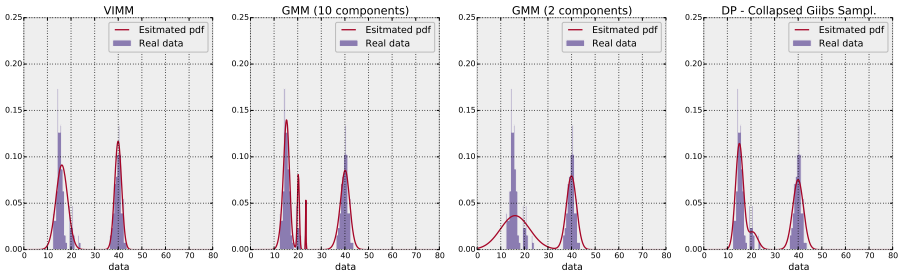


Figure 10.5: Fitting performance - three non separated Gaussian distributions.

	VIMM	GMM-10	GMM-2	DPMM
First dataset	0.154	0.052	0.008	23.94
Second dataset	0.156	0.034	0.011	21.35
Third dataset	0.124	0.067	0.031	30.19

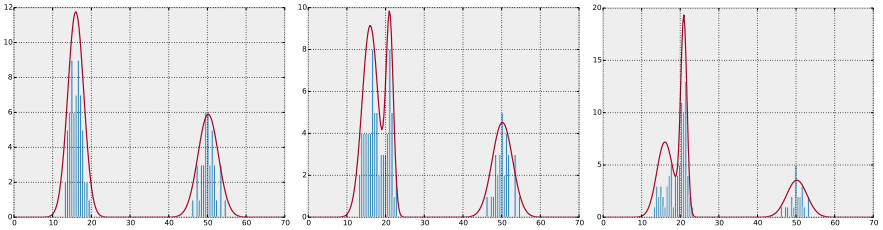
Table 10.1: Time performance of the different models in seconds.

time performance due to their simplicity. Our method is 2 to 5 times slower than conventional GMMs. However, it is much faster than the DPMM that uses collapsed Gibbs sampling to fit the data. The DPMM model is more than 135 times slower than our model.

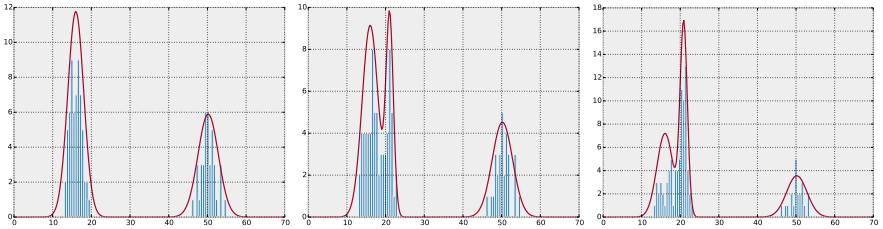
10.7.2 *Updating Mechanism Performance*

In this section we evaluate the quality of the proposed updating mechanism with and without the exploitation of observed data by comparing it against the updating mechanism presented in (Zivkovic, 2004).

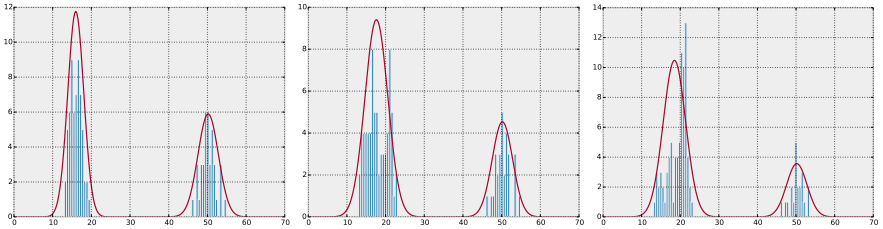
Fig. 10.6 presents the adaptation of our model to new observations (Fig. 10.6(a) and Fig. 10.6(b)) and the model presented in (Zivkovic, 2004) (Fig. 10.6(c)). To evaluate the quality of the adaptation of the models, we used a toy dataset with 100 observations. Observed data were generated from two Gaussian distributions with mean values 16 and 50 and standard deviations 1.5 and 2.0 respectively. The initially trained models are presented in the left column. Then, we generated 25 new samples from a third Gaussian distribution with mean value 21 and standard deviation 1.0. Our model, either it uses the observed data or not, creates a new component and successfully fits the data. On the contrary, the model of (Zivkovic, 2004) is not able to capture the statistical relations of the new observations and fails to separate the data generated from distributions with mean values 16 and 21 (middle column). The quality of presented updating mechanism becomes more clear in the right column, which presents the adaptation of the models after 50 new observations.



(a) Proposed adaptation process using observed data.



(b) Proposed adaptation process without using observed data.



(c) Adaptation process presented in (Zivkovic, 2004).

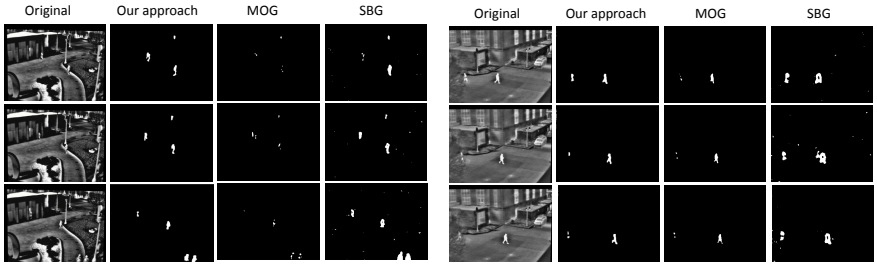
Figure 10.6: Performance evaluation of model updating mechanisms.

10.7.3 *Background Subtraction Algorithm Evaluation*

For evaluating our algorithm, we used the Ohio State University (OSU) thermal datasets and an dataset captured at Athens International Airport (AIA) during a European funding project. OSU datasets contain frames that have been captured using a thermal camera and have been converted to grayscale images. In contrast, the AIA dataset contains raw thermal frames whose pixel values correspond to the real temperature of objects.

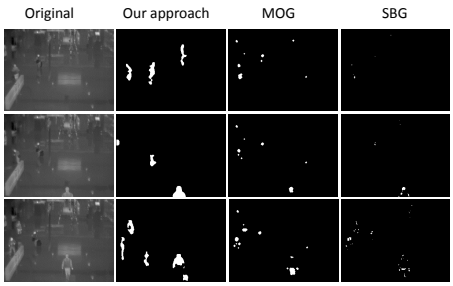
OSU datasets (Davis and Sharma, 2005, 2004, 2007) are widely used for benchmarking algorithms for pedestrian detection and tracking in infrared imagery. Videos were captured under different illumination and weather conditions. AIA dataset was captured using a Flir A315 camera at different Airside Corridors and the Departure Level. Totally, 10 video sequences were captured, with frame dimensions 320×240 pixels of total duration 32051 frames, at 7.5fps, that is, about 1h and 12mins.

We compared our method with method presented by Zivkovic in (Zivkovic, 2004) (MOG), which is one of the most robust and widely used background subtraction technique, and with the method for extracting the regions of interest presented in (Davis and Sharma, 2004, 2007) (SBG) used for thermal data. To conduct the comparison we utilized the objective metrics of *recall*, *precision* and *F1 score* on a pixel wise manner. Figures 10.7 visually present the performance of the three methods. As is observed, our method outperforms both MOG and SBG on all datasets. While MOG and SBG per-



(a) OSU 1 dataset

(b) OSU 2 dataset



(c) AIA dataset

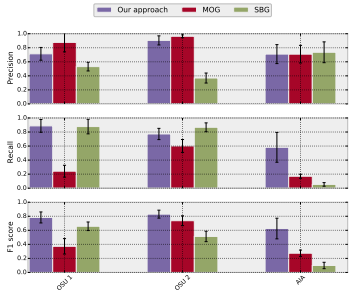
Figure 10.7: Visual results for all datasets.

form satisfactory on grayscale frames of OSU datasets, their performance collapses when they applied on AIA dataset, which contains actual thermal responses. Regarding OSU datasets, MOG algorithm while presents high precision it yields very low recall values, i.e. the pixels that have been classified as foreground are indeed belong to the foreground class, but a lot of pixels that in fact belong to background have been misclassified. SBG algorithm seems to suffer by the opposite problem. Regarding AIA dataset, our method significantly outperforms both methods. In particular, while MOG and SBG algorithms present relative high precision, their recall values are under 0.2. Figure 10.8(a) presents average precision, recall and F1 score per dataset and per algorithm for all frames examined to give an objective evaluation. In Figure 10.8(b) presents the best and worst case in terms of precision, recall and F1 score among all frames examined.

Regarding computational cost, the main load of our algorithm is in the implementation of EM optimization. In all experiments conducted, the EM optimization converges within 10 iterations. Practically, the time required to apply our method is similar to the time requirements of Zivkovic's method making it suitable for real-time applications.

10.8 CONCLUSIONS

In this work a novel algorithm for background subtraction was presented which is suitable for in-camera acceleration in thermal imagery. The presented scheme through an automated parameter estimation process, takes into account the special characteristics of thermal data, and gives highly accurate results without any fine-tuning from the user.



(a) Average performance

			OSU 1	OSU 2	AIA
Our approach	Precision	Worst	0.49	0.75	0.42
		Best	0.88	0.97	0.84
	Recall	Worst	0.54	0.65	**0.25**
		Best	**0.96**	0.95	**0.98**
	F1 score	Worst	**0.60**	**0.74**	**0.31**
		Best	**0.88**	**0.94**	**0.87**
MOG	Precision	Worst	**0.52**	**0.92**	**0.44**
		Best	**0.99**	**0.99**	0.84
	Recall	Worst	0.12	0.46	0.12
		Best	0.39	0.78	0.21
	F1 score	Worst	0.20	0.62	0.19
		Best	0.56	0.87	0.33
SBG	Precision	Worst	0.41	0.23	0.42
		Best	0.66	0.75	**0.90**
	Recall	Worst	**0.56**	**0.78**	0.02
		Best	**0.96**	**0.99**	0.10
	F1 score	Worst	0.53	0.36	0.05
		Best	0.78	0.59	0.16

(b) Best/worst cases

Figure 10.8: Algorithms performance per dataset.

10.9 APPENDIX:DERIVATION OF OPTIMAL VARIATIONAL DISTRIBUTIONS

Using (10.8) and (10.9) the logarithm of $q^*(\boldsymbol{Z})$ is given by

$$\begin{aligned}\ln q^*(\boldsymbol{Z}) =& \mathbb{E}_{\boldsymbol{\varpi}}[\ln p(\boldsymbol{Z}|\boldsymbol{\varpi})]+ \\ &+ \mathbb{E}_{\boldsymbol{\mu},\boldsymbol{\tau}}[\ln p(\boldsymbol{X}|\boldsymbol{Z},\boldsymbol{\mu},\boldsymbol{\tau})] + \mathcal{C}\end{aligned} \tag{10.34}$$

substituting (10.5) and (10.6) into (10.34) we get

$$\begin{aligned}\ln q^*(\boldsymbol{Z}) = \sum_{n=1}^{N}\sum_{k=1}^{K} z_{nk}\Big(&\mathbb{E}\big[\ln \varpi_k\big] + \frac{1}{2}\mathbb{E}\big[\ln \tau_k\big] - \\ &- \frac{1}{2}\ln 2\pi - \frac{1}{2}\mathbb{E}_{\boldsymbol{\mu},\boldsymbol{\tau}}\big[(x_n-\mu_k)^2\tau_k\big]\Big) + \mathcal{C} \Rightarrow\end{aligned}$$

Using (10.9) and (10.8) the logarithm of $q^*(\boldsymbol{\varpi},\boldsymbol{\mu},\boldsymbol{\tau})$ is

$$\begin{aligned}\ln q^*(\boldsymbol{\varpi},\boldsymbol{\mu},\boldsymbol{\tau}) =& \mathbb{E}_{\boldsymbol{Z}}\big[\ln p(\boldsymbol{X}|\boldsymbol{Z},\boldsymbol{\mu},\boldsymbol{\tau})+ \\ &+ \ln p(\boldsymbol{Z}|\boldsymbol{\varpi})+ \\ &+ \ln p(\boldsymbol{\varpi}) + \ln p(\boldsymbol{\mu},\boldsymbol{\tau})\big] + \mathcal{C} = \qquad (10.36a)\\ &= \sum_{n=1}^{N}\sum_{k=1}^{K}\mathbb{E}\big[z_{nk}\big]\ln\mathcal{N}(x_n|\mu_k,\tau_k^{-1})+ \\ &+ \mathbb{E}_{\boldsymbol{Z}}\big[\ln p(\boldsymbol{Z}|\boldsymbol{\varpi})\big] \\ &+ \ln p(\boldsymbol{\varpi}) + \sum_{k=1}^{K}\ln p(\mu_k,\tau_k) + \mathcal{C} \qquad (10.36b)\end{aligned}$$

Due to the fact that there is no term in (10.36b) that contains parameters from both sets $\{\boldsymbol{\varpi}\}$ and $\{\boldsymbol{\mu},\boldsymbol{\tau}\}$, the distribution $q^*(\boldsymbol{\varpi},\boldsymbol{\mu},\boldsymbol{\tau})$ can be factorized as $q(\boldsymbol{\varpi},\boldsymbol{\mu},\boldsymbol{\tau}) = q(\boldsymbol{\varpi})\prod_{k=1}^{K} q(\mu_k,\tau_k)$. The distribution for $q^*(\boldsymbol{\varpi})$ is derived using only those terms of (10.36b) that depend on the variable $\boldsymbol{\varpi}$. Therefore the logarithm of $q(\boldsymbol{\varpi})$ is given by

$$\begin{aligned}\ln q^*(\boldsymbol{\varpi}) &= \mathbb{E}_{\boldsymbol{Z}}\big[\ln p(\boldsymbol{Z}|\boldsymbol{\varpi})\big] + \ln p(\boldsymbol{\varpi}) + \mathcal{C} = \qquad (10.37a)\\ &= \sum_{k=1}^{K}\ln \varpi_k^{(\sum_{n=1}^{N} r_{nk}+\lambda_0-1)} + \mathcal{C} = \qquad (10.37b)\\ &= \sum_{k=1}^{K}\ln \varpi_k^{(N_k+\lambda_0-1)} + \mathcal{C} \qquad (10.37c)\end{aligned}$$

We have made use of $\mathbb{E}[z_{nk}] = r_{nk}$, and we have denote as $N_k = \sum_{n=1}^{N} r_{nk}$. (10.37c) suggests that $q^*(\boldsymbol{\varpi})$ is a Dirichlet distribution with hyperparameters $\boldsymbol{\lambda} = \{N_k + \lambda_0\}_{k=1}^{K}$.

Using only those terms of (10.36b) that depend on variables $\boldsymbol{\mu}$ and $\boldsymbol{\tau}$, the logarithm of $q^*(\mu_k, \tau_k)$ is given by

$$\begin{aligned}
\ln q^*(\mu_k, \tau_k) = &\ln \mathcal{N}(\mu_k | m_0, (\beta_0 \tau_k)^1) + \\
&+ \ln Gam(\tau_k | a_0, b_0) + \\
&+ \sum_{n=1}^{N} \mathbb{E}[z_{nk}] \ln \mathcal{N}(x_n | \mu_k, \tau_k^{-1}) + \mathcal{C} = \\
= &-\frac{\beta_0 \tau_k}{2} (\mu_k - m_0)^2 + \frac{1}{2} \ln(\beta_0 \tau_k) + \\
&+ (a_0 - 1) \ln \tau_k - b_0 \tau_k - \\
&- \frac{1}{2} \sum_{n=1}^{N} \mathbb{E}[z_{nk}] (x_n - \mu_k)^2 \tau_k + \\
&+ \frac{1}{2} \left(\sum_{n=1}^{N} \mathbb{E}[z_{nk}] \right) \ln(\beta_0 \tau_k) + \mathcal{C}
\end{aligned} \tag{10.38}$$

For the estimation of $q^*(\mu_k | \tau_k)$, we use (10.38) and keep only those factors that depend on μ_k.

$$\begin{aligned}
\ln q^*(\mu_k | \tau_k) = &-\frac{\beta_0 \tau_k}{2} (\mu_k - m_0)^2 - \\
&- \frac{1}{2} \sum_{n=1}^{N} \mathbb{E}[z_{nk}] (x_n - \mu_k)^2 \tau_k = && (10.39a) \\
= &-\frac{1}{2} \mu_k^2 \Big(\beta_0 + N_k \Big) \tau_k + \\
&+ \mu_k \tau_k \Big(\beta_0 m_0 + N_k \bar{x}_k \Big) + \mathcal{C} \Rightarrow && (10.39b) \\
q^*(\mu_k | \tau_k) = &\mathcal{N}(\mu_k | m_k, (\beta_k \tau)^{-1}) && (10.39c)
\end{aligned}$$

where $\bar{x}_k = \frac{1}{N_k} \sum_{n=1}^{N} r_{nk} x_n$, $\beta_k = \beta_0 + N_k$ and $m_k = \frac{1}{\beta_k} (\beta_0 m_0 + N_k \bar{x}_k)$.

After the estimation of $q^*(\mu_k | \tau_k)$, logarithm of the optimized the distribution $q^*(\tau_k)$ is given by

$$\begin{aligned}
\ln q^*(\tau_k) = &\ln q^*(\mu_k, \tau_k) - \ln q^*(\mu_k | \tau_k) = && (10.40a) \\
= &\left(a_0 + \frac{N_k}{2} - 1 \right) \ln \tau_k - \\
&- \frac{1}{2} \tau_k \Big(\beta_0 (\mu_k - m_0)^2 + \\
&+ 2 b_0 + \sum_{n=1}^{N} r_{nk} (x_n - \mu_k)^2 - \\
&- \beta_k (\mu_k - m_k)^2 \Big) + \mathcal{C} \Rightarrow && (10.40b) \\
q^*(\tau_k) = &Gam(\tau_k | a_k, b_k) && (10.40c)
\end{aligned}$$

The parameters a_k and b_k are given by

$$a_k = a_0 + \frac{N_k}{2} \tag{10.41a}$$

$$b_k = b_0 + \frac{1}{2}\left(N_k \sigma_k + \frac{\beta_0 N_k}{\beta_0 + N_k} \left(\bar{x}_k - m_0\right)^2 \right) \tag{10.41b}$$

where $\sigma_k = \frac{1}{N_k} \sum_{n=1}^{N} (x_n - \bar{x}_k)^2$.

11

DEEP LEARNING BASED HYPERSPECTRAL DATA CLASSIFICATION

11.1 MOTIVATION

Recent advances in optics and photonics have allowed the development of hyperspectral imaging sensors with higher spectral and spatial resolution onboard various satellite, aerial, UAV and ground acquisition platforms. The efficient exploitation of finer spatial and spectral information can ameliorate significantly material detection and object recognition applications by revealing and modeling the subtle differences in spectral signatures of various objects.

Recognizing various materials, objects and terrain land cover classes based on their reflectance properties can be viewed as a classification task i.e., classify image pixels based on their spectral characteristics. Although, hyperspectral imaging have been used in a wide variety of applications, such as agriculture, surveillance, astronomy and biomedical imaging (Chang, 2013), it has its own unique challenges including; i) high dimensional data, ii) limited number of labeled samples and iii) large spatial variability of spectral signatures (Camps-Valls and Bruzzone, 2005).

Most of the existing work towards the classification of hyperspectral data, follow the conventional paradigm of pattern recognition, which consists of two separate steps. Firstly, complex handcrafted features are computed from the raw data. Then, the obtained features are used to learn classifiers, such as Support Vector Machines (SVM) (Li and Du, 2015; Camps-Valls and Bruzzone, 2009), Gaussian Process models (Ying Yang et al., 2015), superpixel graphical probabilistic models (Zhan et al., 2015) and Neural Networks (NN) (Chen et al., 2014). In particular, for high dimensional data and when few training samples are available statistical learning methods have been employed to tackle the high dimensionality and heterogeneity of hyperspectral data (Camps-Valls et al., 2014).

However, due to the high diversity of depicted materials, it is rarely known which features are important for the classification task. In contrast to the conventional paradigm of pattern recognition, deep learning models (LeCun et al., 1998; Hinton and Salakhutdinov, 2006b; Hinton et al., 2006; Bengio et al., 2007) are a class of machines that can learn hierarchies of features by building high level features from low level ones, thereby automating the process of feature construction. Furthermore, for bigger datasets and quite large images with very high spatial and spectral resolution, deep learning frameworks seems to fit and address more effectively the classification problem (Chen et al., 2014; Makantasis et al., 2015). Techniques based on deep learning have already shown promising results both for the detection of particular objects, like man-made ones (Mnih and Hinton, 2012) or vehicles (Orr and Müller, 2003) and for the classification of hyperspectral data (Chen et al., 2014; Makantasis et al., 2015).

More specifically, a deep learning framework was employed in (Chen et al., 2014) for the classification of hyperspectral data with quite promising results. In particular, Stacked Autoencoders (SAEs) have been used as build-

ing blocks and the concept of greedy layer-wise pre-training is adopted to construct a deep architecture that hierarchically builds high level spectral features for each pixel. Spectral features were combined in a separate step with spatially-dominated information and are fed as input to a logistic regression classifier.

11.2 OUR CONTRIBUTION

We propose a deep learning framework for the classification of hyperspectral data into multiple classes. In particular, we propose the exploitation of a modified Convolutional Neural Network (CNN)(LeCun et al., 1998), which conducts the task of high level features construction and a Multi-Layer Perceptron (MLP), which is responsible for the classification task. We call the proposed deep learning model as DL-CNN.

In contrast to (Chen et al., 2014), where two separate are adopted as explained above, our approach is based on a *unified* framework combining spectral and spatial information in a *single* step to reduce the computational cost. Furthermore, our model requires no pre-training for hierarchically constructing high level features. The drawback of pre-training, as adopted in SAEs (this is also the case of (Chen et al., 2014)), is that the model can learn the identity function and thus to potentially propagate the same features from layer to layer resulting in a failure of hierarchical feature construction process. On the contrary, CNNs construct such features during the training phase when their parameters are being fine-tuned. By overpassing the requirement for pre-training, CNNs overcome this drawback.

To summarize, by exploiting CNNs and MLPs, the developed system (DL-CNN) i) hierarchically constructs high level spectral-spatial features at once, in a single step, ii) avoids the propagation of features from layer to layer and iii) achieves low cost predictions due to the feed forward nature of CNNs and MLPs.

11.3 THE DEEP LEARNING PARADIGM CNNS

11.3.1 *Deep Learning*

Artificial Neural Networks (ANN) are rooted in a classical theorem by Kolmogorov (Kolmogorov, 1963), which states that every continuous function f on $[0,1]^d$ can be written as summands of continuous functionals whose form depends on f. ANNs utilize bounded, continuous and increasing functionals and allow the number of tunable parameters to be high enough such that any continuous function is approximated.

Despite its theoretical importance, this theorem is very hard to be applied on real world problems. The approximation accuracy of an unknown function; *i.e.*, performance on training set, is proportional to the number of tunable parameters. However, the more relevant measure is the performance of the system on a set of unseen examples, which is called the test set; *i.e.*, generalization ability of the learning machine. Unfortunately, the generalization ability of ANNs to unseen examples is inversely proportional to the number of network's tunable parameters (Vapnik, 1995, 1998).

Due to this fact, people could not harness powerful ANNs with multiple hidden layers, *i.e.*, deep networks, until, the publication of the seminal paper by Yann LeCun *et al.* (LeCun et al., 1998), which introduces the idea of

tied weights in order to implement a deep network and keep the number of tunable parameters low, and the introduction by Hinton and Salakhutdinov of layer-wise network initialization via the *unsupervised pre-training* concept (Hinton and Salakhutdinov, 2006b).

Till then, deep learning models have been widely used in a variety of applications and seem to outperform conventional swallow machines (Sutskever and Hinton, 2008; Le Roux and Bengio, 2010). Typical deep learning architectures include Deep Belief Networks (Hinton et al., 2006), Convolutional Neural Networks (CNN) (Krizhevsky et al., 2012; Simard et al., 2003), Deep and Restricted Boltzmann Machines (Salakhutdinov et al., 2007; Salakhutdinov and Hinton, 2009b) and networks based on Stacked Autoencoders (SAE) (Vincent et al., 2008, 2010; Shin et al., 2013). All these architectures, by propagating input information from one layer to another are able to *hierarchically* construct high level features in a fully *automated* way.

11.3.2 *Convolutional Neural Networks*

CNNs consist a type of deep learning machine that relies as much as possible on learning in the feature constructor itself. In other words, CNNs hierarchically construct high level features, in a automated way, that are appropriate for a specific classification task.

The architecture of CNNs employs three concrete ideas; a) *local receptive fields*, b) *tied weights* and c) *spatial sub-sampling*. Based on local receptive field, each unit in a convolutional layer receives inputs from a set of neighboring units belonging to the previous layer. This way neurons are capable of extracting elementary visual features such as edges or corners. These features are then combined by the subsequent convolutional layers in order to detect higher order features.

Furthermore, the idea that elementary feature detectors, which are useful on a part of an image, are likely to be useful across the entire image, is implemented by the concept of tied weights. The concept of tied weights constraints a set of units to have identical weights. Concretely, the units of a convolutional layer are organized in planes. All units of a plane share the same set of weights. Thus, each plane is responsible for constructing a specific feature. The output of planes are called feature maps. Each convolutional layer consists of several planes, so that multiple feature maps can be constructed at each location.

During the construction of a feature map, the entire image is scanned by a unit whose states are stored at corresponding locations in the feature map. This construction is equivalent to a convolution operation, followed by an additive bias term and sigmoid function

$$\boldsymbol{y}^{(d)} = \sigma(\boldsymbol{W}\boldsymbol{y}^{(d-1)} + \boldsymbol{b}), \tag{11.1}$$

where d stands for the depth of the convolutional layer, $\boldsymbol{W}$ is the weight matrix, $\boldsymbol{b}$ is the bias term. For fully connected neural networks, the weight matrix is full, *i.e.* connects every input to every unit with different weights.

For CNNs, the weight matrix $\boldsymbol{W}$ is very sparse due to the concept of tied weights. Thus, $\boldsymbol{W}$ has the form of

$$\boldsymbol{W} = \begin{bmatrix} \boldsymbol{w} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \boldsymbol{w} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \cdots & \mathbf{0} & \boldsymbol{w} \end{bmatrix}, \tag{11.2}$$

where $\boldsymbol{w}$ are matrices having the same dimensions with the units' receptive fields. Employing a sparse weight matrix reduces the number of network's tunable parameters and thus increases its generalization ability. Multiplying $\boldsymbol{W}$ with layer inputs is like convolving the input with $\boldsymbol{w}$, which can be seen as a *trainable filter*. If the input to $d-1$ convolutional layer is of dimension $N \times N$ and the receptive field of units at a specific plane of convolutional layer d is of dimension $m \times m$ then the constructed feature map will be a matrix of dimensions $(N-m+1) \times (N-m+1)$. Specifically, the element of feature map at (i,j) location will be

$$\boldsymbol{y}_{ij}^{(d)} = \sigma(\boldsymbol{x}_{ij}^{(d)} + b) \tag{11.3}$$

with

$$\boldsymbol{x}_{ij}^{(d)} = \sum_{a=0}^{m-1} \sum_{b=0}^{m-1} \boldsymbol{w}_{ab} \boldsymbol{y}_{(i+a)(j+b)}^{(d-1)}, \tag{11.4}$$

where the bias term b is scalar. Using equations (11.4) and (11.3) sequentially for all (i,j) positions of input, the feature map for the corresponding plane is constructed.

11.4 CNNS FOR HYPERSPECTRAL DATA CLASSIFICATION

We consider the exploitation of a deep learning architecture for the classification of hyperspectral data, i.e. the classification of each pixel to a predefined number of classes based on their spectral and spatial properties. The spectral characteristics are associated with the reflectance properties at every pixel for every spectral band, while spatial information is derived by taking into consideration its neighbors.

Towards this direction, high-level features that encode pixels' spectral and spatial information, are hierarchically constructed using a CNN. Although, it has been shown that CNNs can achieve superior performance on visual recognition tasks without relying on handcrafted features, they are suitable for pixel-based classification tasks due to the fact that they produce global image features.

A hyperspectral image is represented as a 3D tensor of dimensions $h \times w \times c$, where h and w correspond to the height and width of the image and c to its channels (spectral bands). In order to be aligned with the specific nature of CNNs (global image feature construction), we have to decompose the captured hyperspectral image into patches, each one of which contains spectral and spatial information for a specific pixel.

More specifically, in order to classify a pixel p_{ij} at location (i,j) on image plane and successfully fuse spectral and spatial information, we use a square patch of size $s \times s$ centered at pixel p_{ij} . Let us denote as l_{ij} the class label of the pixel at location (i,j) and as w_{ij} the patch centered at pixel p_{ij}. Then, we

can form a dataset $D = \{w_{ij}, l_{ij}\}$ for $i = 1, 2, \cdots, w$ and $j = 1, 2, \cdots, h$. Patch w_{ij} is also a 3D tensor with dimension $s \times s \times c$, which contains spectral and spatial information for the pixel located at (i, j).

Moreover, tensor w_{ij} is divided into c matrices of dimensions $s \times s$ which are fed as input into a CNN to hierarchically build high-level features that encode spectral and spatial characteristics of pixel p_{ij}. Then, these features are fed to a Multi Layer Perceptron (MLP), which is responsible for the classification task. At this point, it has to be mentioned that after the completion of the training phase our deep learning model is capable of classifying patches and not pixels. We assume that the label of the patch centered at location (i, j) on image plane must be the same with the pixel at the same location. Although, it is a strong assumption, for this specific problem at hand, it is valid for the vast majority of the pixels due to the fact that neighboring pixels is very probable to belong to the same class.

In the following, we describe the architecture of the proposed system. Firstly, the proposed approach for the dimensionality reduction of the raw input data is presented and then the structures of the CNN and the MLP are given.

11.4.1 *Reducing the Dimension of Raw Input Data*

In contrast to RGB images that consist of three color channels, the hundreds of channels (network inputs) along the spectral dimension of a hyperspectral image may increase the computational cost of training and prediction phases to non acceptable levels. However, the spectral signature of a material is very specific. Through a simple statistical analysis of pixels' spectral responses we observed two different things. Firstly, the variance of spectral responses of pixels that depict the same material is very small, and secondly, pixels that depict different materials, either they respond to different spectral bands or when they respond to the same spectral bands the values of their responses is very divergent. These observations suggest that redundant information is present along the spectral dimension of a hyperspectral image. Therefore, a dimensionality reduction technique can be employed to reduce the dimensionality of the raw input data in order to speed up the training and prediction processes.

For dimensionality reduction, Randomized Principal Component Analysis (R-PCA) (Halko et al., 2009) is introduced along the spectral dimension to condense the whole image. Principal Component Analysis (PCA) projects data to a lower dimensional space that preserves most of the variance by dropping components associated with lower eigenvalues. R-PCA limits the computation to an approximate estimate of principal components to perform data transformation. Thus, it is much more computationally efficient than PCA and suitable for large scale datasets, like hyperspectral images. Due to the approximate estimation of principal components, R-PCA is less accurate than PCA, however, the authors of (Halko et al., 2009) prove strong bounds on the quality of this approximation suggesting no significant deterioration of the quality.

It should be noted that the aforementioned step does cast away spectral information, but since R-PCA is applied along the spectral dimension, the spatial information remains intact. The number of principal components that are retained after the application of R-PCA, is appropriately set, in order to keep at least 99.9% of initial information. This is very important, since

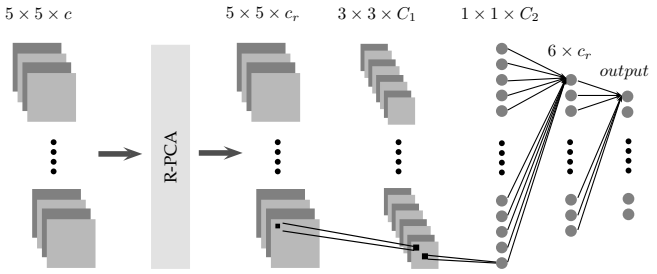


Figure 11.1: Overall system architecture.

dimensionality reduction is conducted by taking into consideration the maximum allowed information loss and not a fixed number of principal components. During the experimentation process on widely used hyperspectral datasets, due to the correlations of pixels' spectral responses, 99.9% of initial information is preserved by using the first 10 to 95 principal components, reducing this way up to 20 times the dimensionality of the raw input.

11.4.2 *Parameter Selection for the DL-CNN*

After dimensionality reduction, each patch is a tensor of dimensions $s \times s \times c_r$. Parameter c_r corresponds to the number of principal components that preserve at least 99.9% of initial information, while the parameter s determines the number of neighbors of each pixel that will be taken into consideration during the classification task.

During experimentation process we set the parameter s to be equal to 5, in order to take into consideration the closest 24 neighbors of each pixel. By increasing the value of s, the number of neighbors that are taken into consideration is increased and thus the computational cost of classification is increased, also. However, setting the parameter s to a value larger than 5, no further improvement on classification accuracy was reported in all experiments. On the contrary, increasing the value of s over 13, deteriorates classification accuracy. This is justified by the fact that our previous assumption, which states that the label of the patch centered at location (i, j) on image plane must be the same with the label of the pixel at the same location, is not valid for large s.

Having estimate the values of the parameters s and c_r , we can proceed with the CNN structure design. The first layer of the proposed CNN is a convolutional layer with $C_1 = 3 \times c_r$ trainable filters of dimension 3×3. This layer delivers C_1 matrices (feature maps) of dimensions 3×3 (during convolution we don't take into consideration the border of the patch). In contrast to conventional CNNs, we do not use a sub-sampling layer after the convolution layer, since we don't take into account any translation and scale invariance. The first convolutional layer employs $(c_r \times C_1 \times 3 \times 3) + (C_1 \times 3 \times 3)$ connections and $(C_1 \times 3 \times 3) + C_1$ weights.

The first convolutional layer is followed by a second convolutional layer with $C_2 = 3 \times C_1$ trainable filters. Again, the filters are 3×3 matrices. The second convolutional layer delivers a vector with C_2 elements, which is fed as input to the MLP classifier. The second convolutional layer employs $(C_1 \times C_2 \times 3 \times 3) + C_2$ connections and $(C_2 \times 3 \times 3) + C_2$ weights.

The number of MLP hidden units is smaller than the dimensionality of its input. In particular, we set the number of hidden units C_h to equal $6 \times c_r$. The

MLP employs $(C_2 \times 6 \times c_r) + (6 \times c_r)$ connections and weights between the input and the hidden layer. Finally, if we denote as C_l the number of classes (number of output units), the MLP employs $(6 \times c_r \times C_l) + C_l$ connections and weights between the hidden layer and the output layer. The overall system architecture is presented in Fig.11.1.

11.4.3 *Fine-tuning and Classification*

For training the deep learning architecture the standard back propagation algorithm (Rumelhart et al., 1985) was employed, in order to learn the optimal model parameters, *i.e.*, minimize the negative log-likelihood of the data sets under the model parameterized by MLP weights and filters elements.

Although back propagation is a well known algorithm, for the sake of completeness, we briefly describe its application on the proposed learning model. Our description regards the error propagation from one convolutional layer to a previous one. We do not describe the error propagation for the MLP part of our learning model, because the application of back propagation on feed forward fully connected networks is a well studied problem (Chauvin and Rumelhart, 1995; Phansalkar and Sastry, 1994; Van Ooyen and Nienhuis, 1992).

Let's denote as L the error at the last convolutional layer, whose depth is d. What we want to estimate is the error at the previous layer and the gradient for each weight. The gradient of weight w_{ab} located at position (a, b) on local receptive field is

$$\begin{aligned} \frac{\partial L}{\partial w_{ab}} &= \sum_{i=0}^{m-1} \sum_{j=0}^{m-1} \frac{\partial L}{\partial x_{ij}^{(d)}} \frac{\partial x_{ij}^{(d)}}{\partial w_{ab}} = \\ &= \sum_{i=0}^{m-1} \sum_{j=0}^{m-1} \frac{\partial L}{\partial x_{ij}^{(d)}} y_{(i+a)(j+b)}^{(d-1)}, \end{aligned} \tag{11.5}$$

we recall that $m \times m$ is the size of the local receptive field. In order to compute the gradient of weight w_{ab} we need to estimate the gradient $\partial L / \partial x_{ij}^{(d)}$. This gradient is equal to

$$\begin{aligned} \frac{\partial L}{\partial x_{ij}^{(d)}} &= \frac{\partial L}{\partial y_{ij}^{(d)}} \frac{\partial y_{ij}^{(d)}}{\partial x_{ij}^{(d)}} = \\ &= \frac{\partial L}{\partial y_{ij}^{(d)}} \frac{\partial \left(\sigma(x_{ij}^{(d)})\right)}{\partial x_{ij}^{(d)}} = \frac{\partial L}{\partial y_{ij}^{(d)}} \sigma'(x_{ij}^{(d)}) \,. \end{aligned} \tag{11.6}$$

Due to the fact that $\partial L / \partial y_{ij}^{(d)}$ is already known by applying error propagation on the feed forward fully connected part of the proposed learning model the gradient $\partial L / \partial x_{ij}^{(d)}$ can be easily computed using (11.6).

In order to propagate the error at the previous convolutional layer, whose depth is $d-1$, we need to estimate the gradient $\partial L/\partial y_{ij}^{(d-1)}$. This gradient can be computed as

$$\begin{aligned}\frac{\partial L}{\partial y_{ij}^{(d-1)}} &= \sum_{i=0}^{m-1}\sum_{j=0}^{m-1}\frac{\partial L}{\partial x_{(i-a)(j-b)}^{(d)}}\frac{\partial x_{(i-a)(j-b)}^{(d)}}{\partial y_{ij}^{(d-1)}} = \\ &= \sum_{i=0}^{m-1}\sum_{j=0}^{m-1}\frac{\partial L}{\partial x_{(i-a)(j-b)}^{(d)}} w_{ab}\,.\end{aligned} \tag{11.7}$$

Using recursively the equations (11.5), (11.6) and (11.7) the error can be propagated to all convolutional layers and the gradients for all network's weights can be computed. For learning the most appropriate network's weights for the given classification task, a gradient based optimization approach is followed exploiting Stochastic Gradient Descent (SGD) (Bottou, 2010; Zinkevich et al., 2010).

After the completion of training phase, the classification takes place by using a soft-max activation function for the output layer of the MLP. The output of the soft-max function can be interpreted as the conditional probabilities of an example to belong to a specific class given the input and network parameters. Mathematically the soft-max function is defined as

$$P(l=i|\boldsymbol{y}^{D-1},\boldsymbol{W}^{out}) = \frac{e^{\boldsymbol{W}_i^{out}\boldsymbol{y}^{D-1}}}{\sum_{j=1}^{C_l} e^{\boldsymbol{W}_j^{out}\boldsymbol{y}^{D-1}}}\,, \tag{11.8}$$

where D denotes the maximum depth of the network. The bias term has been concatenated with output layer's weights into a single unified matrix $\boldsymbol{W}^{out}$. The soft-max function ensures that $0 \leq P(l=i|\boldsymbol{y}^{D-1},\boldsymbol{W}^{out}) \leq 1$ and $\sum_i P(l=i|\boldsymbol{y}^{D-1},\boldsymbol{W}^{out}) = 1$ for $i = 1, 2, ..., C_l$.

A flowchart which describes the main algorithmic steps of the developed deep learning classifier (*DL-CNN*) is given in CL-CNN Algorithm.

11.5 EXPERIMENTAL RESULTS AND VALIDATION

11.5.1 *Dataset Description and Experimental Setting*

The proposed deep learning classifier was evaluated using six widely used and public available hyperspectral datasets acquired from the AVIRIS and ROSIS sensors. In particular, we employed the

- *Indian Pines* dataset, which depicts a test site in North-western Indiana and consists of 145×145 pixels and 224 spectral reflectance bands in the wavelength range 0.4 to 2.5×10^{-6} meters. This dataset depicts materials that belong to 16 different classes.
- *Salinas* Valley, California, 224-band hyperspectral image, which consists of 512×217 pixels that belong to 16 different classes.
- *Pavia Centre* and *Pavia University* datasets, whose number of spectral bands are 103 and 102 respectively. Pavia dataset consists of 1096×1096 pixels, while Pavia University dataset consists of 610×610 pixels. Both datasets depict pixels that belong to 9 different classes.

***DL-CNN* Algorithm**: Deep Learning-based hyperspectral data classification with CNN

```
load hyperspectral data
reduce dimension using R-PCA (Section 11.4.1))
split hyperspectral data into overlapping windows
    form dataset for training (Section 11.4)
initialize network architecture (Section 11.4.2)
    number of feature maps at each convolutional layer
    size of receptive fields for each feature map
    number of units of MLP hidden layer
    number of classes
    learning rate
    number of mini batches
    number of training epochs
initialize network parameters
start training phase
    while current epoch ≤ number of epochs
      while current mini batch ≤ number of mini batches
        if stopping criteria are satisfied
          stop training
        else
          use back propagation to compute network's
           weights and biases gradients
          use SGD to update biases and weights
```

- *Kennedy Space Center (KSC)* dataset, which consists of 224 spectral bands. After removing water absorption and low SNR bands, 176 bands were used for analysis. KSC dataset consists of 512×614 pixels and depicted materials belong to 13 different classes.
- *Botswana* dataset, which consists of 242 spectral bands. After the removal of the uncalibrated and noisy bands (UT Center for Space Research) 145 bands were used, while the size of each image was 256×1476 pixels and the ground truth indicated 14 different classes.

Experiments were organized into two parts. The first part aims at analyzing the effectiveness of the developed (*DL-CNN*) deep learning architecture. The performance of the proposed model was compared against deep learning architectures based on the exploitation of Stacked Autoencoders (*DL-SAE*), like the one presented in (Chen et al., 2014), and against SVM-based methods that utilize linear and RBF kernels. In order to compare and quantify the performance of the developed model, several comparisons were conducted in terms of classification accuracy and time requirements.

The second part aims at evaluating the performance of the developed system against state-of-the-art hyperspectral data classification techniques that employ swallow architectures.

For evaluating the classification performance of the proposed system, we split each one of the datasets into three parts; *i.e.* training, validation and testing sets. The assessment of the classification performance was based on the calculated overall accuracy, which corresponds to the number of misclassified examples. Furthermore, we investigate the generalization capabilities of the developed system to unseen examples and varying size datasets. For this reason, we used different splitting ratios for creating training, validation and testing sets. Specifically, we used five different splitting ratios, letting the training set to vary from 5% to 80% of the size of the entire dataset. Finally, training, validation and testing sets were created by *randomly* selecting the appropriate number of examples from the whole dataset.

Last but not least, during the second part of the experiments we compared our method (*DL-CNN*) against state-of-the-art techniques based on Gaussian Process Models (Ying Yang et al., 2015), Multilayer Superpixel Graph and Loopy Belief Propagation (Zhan et al., 2015) and Adaptive Sparse Representation (Li and Du, 2015).

11.5.2 *Assessing the performance of the developed DL-CNN*

During the first step of the developed method the application of R-PCA on raw hypercubes is performed in order to condense the raw data along the spectral dimension (Section 11.4.1). These principal components retain the 99.9% of the initial information. Their number along with the number of classes of each dataset, determine the architecture of the *DL-CNN* classifier (Section 11.4.2). Table 11.1 presents the number of the calculated principal components c_r for each dataset along with the number of feature maps (C_1 and C_2) of each convolutional layer and the numbers of units (C_h and C_l) of the hidden and output layers of the MLP.

We compared our model against another deep learning classifier, which is based on the exploitation of Stacked Autoencoders *DL-SAE* (Chen et al., 2014). To develop an architecture based on Autoencoders we followed the approach presented in (Chen et al., 2014). According to this approach the spectral information of each pixels remains intact and corresponds to raw pixels' spectral signatures, while pixels' spatial information is derived by

	c_r	C_1	C_2	C_h	C_l
Pavia Univesrity	10	30	90	60	9
Pavia Centre	15	45	135	90	9
KSC	95	285	855	570	13
Botswana	45	135	405	270	14
Indian Pines	70	210	630	420	16
Salinas	10	30	90	60	16

Table 11.1: Number of principal components and learning architecture parameters

extracting the first several principle components of their neighbors (the authors of (Chen et al., 2014) suggest to use the closest 24 neighbors of each pixel). For classification purposes each pixel is described by a feature vector, which is created by concatenating pixel's spectral signature and spatial information, which has been flattened to one dimension. In order to conduct a fair comparison, the number of principal components that were used to encode spatial information is set to be the same as the number of components that were used by the developed method (*DL-CNN*). For all datasets we used four auto-encoders *i.e.* four hidden layers, while the number of units at each hidden layer was sequentially decreasing, in order to hierarchically construct high level features and at the same time to avoid learning the identity function during pre-training.

Furthermore, the developed method was compared against SVM-based approaches that use Radial Basis Function (RBF) and linear kernels. In order to conduct a fair comparison the employed SVM classifier should have been able to exploit pixels' spectral and spatial information during the training phase. For this reason each pixel was represented by its own spectral responses as well as the responses of its 24 closest neighbors. To this end, a pixel at location (i, j) on image plane was represented by the spectral responses of a patch centered at the same location. Although this representation is a 3D tensor, it was flattened to form a 1D vector, in order to be utilized for training the SVMs.

Table 11.5 summarizes the performance of all classifiers for all datasets. Both classifiers (*DL-CNN* and *DL-SAE*) that follow the deep learning paradigm outperform conventional SVM-based classifiers almost for all datasets and all splitting ratios. This emphasize the importance of hierarchical high level feature construction through deep learning architectures.

The developed *DL-CNN* classifier presented the higher classification accuracy rates in all datasets and splitting ratios against the *DL-SAE* and the SVM-based ones. In particular, for the *Pavia University*, *KSC*, *Botswana* and *Indian Pines* datasets with small ratios (*e.g., 5%*) the developed *DL-CNN* method managed to increase the overall accuracy rates by more than 7% surpassing by a significant margin the recently proposed *DL-SAE* algorithm (Chen et al., 2014). When the size of the training set is small, due to the employed concept of tied weights that limits the total number of network's tunable parameters, our model presents much higher classification accuracy. This fact highlight the robust generalization abilities of the developed method to unseen examples, even for small datasets and training samples.

In Fig. 11.2, we examine the classification accuracy from a visual perspective. Pixels, corresponding to annotated and not-annotated regions, for each one of the datasets where classified using the developed *DL-CNN* deep learning approach. The resulted classification maps for 5% (left) and 80% (middle) splitting ratios along with the ground truth (right) are shown for all datasets. After a close look, one can observe that by fusing spectral and spatial information the classification process results to the detection of compact areas, eliminating noisy scatter points. Moreover, in most cases (apart from the *Indian Pines* and *KSC*) datasets even with a small number of training samples (*e.g., 5%*) the *DL-CNN* algorithm managed to approximate the classification map of the one resulting with an *80%* splitting ratio. This observation is also verified by the quantitative evaluation (Table 11.5) which indicates that for these particular cases the increase in the OA rates is less than 7% for splitting ratios from 5% to 80%.

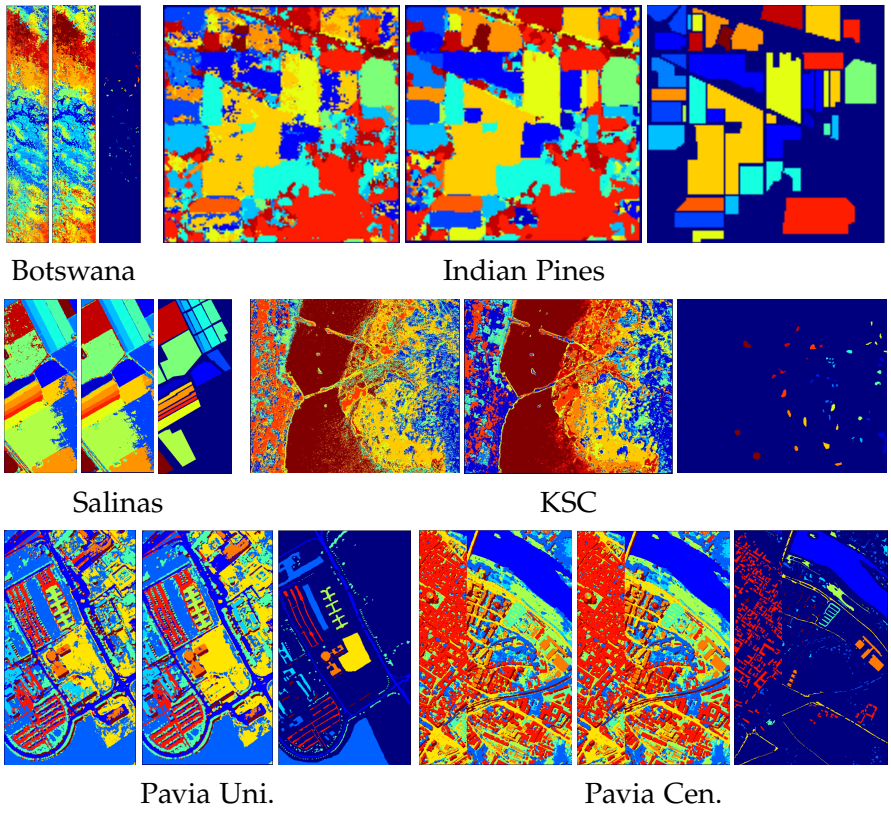


Figure 11.2: Visualization of classification accuracy for al datasets and splitting ratios 5% and 80%. The right image for each dataset represents its ground truth, the middle 80% splitting ratio and left 5% splitting ratio.

Finally, in Fig.11.3 the misclassification error is presented in regard to the number of training epochs for the Pavia Centre and Indian Pines datasets. During the training process we used Stochastic Gradient Descent (SGD) with 500 mini batches. These two diagrams suggest that the training process for the proposed system converges in almost 20 epochs. Therefore, early stopping criteria can be consider during the training procedure, in order to reduce computational cost, without deteriorating classification performance.

11.5.3 *Comparison against Gaussian Process Models (Ying Yang et al., 2015)*

Recently in (Ying Yang et al., 2015) Gaussian Process Models were introduced employing different kernel function to tackle hyperspectral data classification. More specifically, linear, polynomial, RBF, ARD, rational quadratic and NN kernels were employed. For the validation authors utilized *Indian Pines*, *Pavia Centre* and *Pavia University* datasets using 200 examples of each class to form the training set, while in the case that the amount of points of any class was less than 200, 50% of its points were selected as training data. In order to conduct a fair comparison we formed the training set in the same way and we compared our model, in terms of overall accuracy. The results

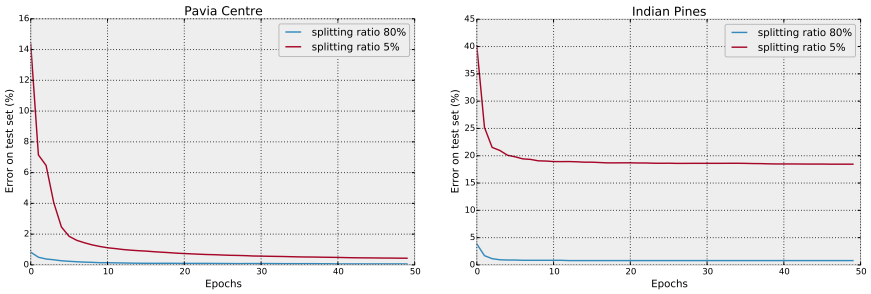


Figure 11.3: Misclassification error on test set in regard to the number of training epochs for Pavia Centre and Indian Pines dataset.

	Indian Pines	Pavia Centre	Pavia Univ.
DDL-CNN	**91.56%**	**99.39%**	**96.60%**
GPMs	87.06%	98.32%	90.87%

Table 11.2: Comparison against Gaussian Process Models of (Ying Yang et al., 2015)

of this comparison are presented in Table 11.2. Our method outperforms the method of (Ying Yang et al., 2015) for all datasets, regardless the kernel function that is used by the Gaussian Process Model. In particular, for the *Indian Pines* and *Pavia University* datasets the increase in the resulting OA after the application of the developed *DL-CNN* is more than 5%.

11.5.4 *Comparison against Multilayer Superpixel Graph and Loopy Belief Propagation (Zhan et al., 2015)*

The developed classifier was, also, compared with the recently proposed classification approach of (Zhan et al., 2015) which is based on multilayer superpixel graph and loopy belief propagation *MSG-LBP*. Authors proposed a graph-based representation and exploited a merging algorithm to generate multiscale superpixels for hyperspectral data. Then, they tackle the classification problem using loopy belief propagation for message passing between superpixels. For validating their method, they utilize *Indian Pines* and *Pavia University* datasets. The training set was formed by setting the splitting ratio equal to 6.7%. For conducting a fair comparison, we formed the set for training our model in the same way and compared both models in terms of overall classification accuracy. The results of this comparison are presented in Table 11.3. The *MSG-LBP* method of (Zhan et al., 2015) outperforms our method for the *Indian Pines* dataset, due to the fact that this dataset contains in the reference/ground truth data many classes with less than 50 pixels. Therefore, by setting the splitting ratio equal to 6.7% certain classes were represented with less than 3 pixels in the training set. For such cases the provided training information is not sufficient for constructing high level features capable to describe such classes. In contrast to *Indian Pines*, the *Pavia University* dataset contains sufficient examples even with low splitting ratios at each class. In this case the developed *DL-CNN* method outperforms the *MSG-LBP* of (Zhan et al., 2015) for more than 5%.

	Indian Pines	Pavia University
DL-CNN	86.54%	**97.46%**
MSG-LBP	**93.06%**	92.32%

Table 11.3: Comparison against *MSG-LBP* (Zhan et al., 2015)

	Indian Pines	Pavia University
DL-CNN	**88.32%**	**92.36%**
ASR	$\simeq$ 85.00%	$\simeq$ 86.00%

Table 11.4: Comparison against the adaptive sparse representation (*ASR*) classifier of (Li and Du, 2015)

11.5.5 *Comparison against the Adaptive Sparse Representation Classifier (Li and Du, 2015)*

Recently, an adaptive sparse representation (*ASR*) for generating discriminative sparse codes was proposed (Li and Du, 2015) which can efficiently represent and contribue to hyperspectral data classification. The validation of the proposed in (Li and Du, 2015) model was performed by utilizing *Indian Pines* and *Pavia University* datasets using 110 examples of each class to form the training set. In order to conduct a fair comparison we form the training set in the same way and we compared both models in terms of overall accuracy. The results of this comparison are presented in Table 11.4. For both datasets, our method outperforms adaptive sparse representation (*ASR*) classifier. In particular, the developed *DL-CNN* algorithm managed to increase the resulting overall accuracy rates by approximately 4% and 7% for the *Indian Pines* and *Pavia University* datasets, respectively.

11.6 CONCLUSIONS

We designed, developed and validated a novel deep learning-based approach for hyperspectral data classification. Following deep learning paradigm, via the exploitation of CNNs and MLPs, our approach hierarchically constructs high-level features that encode pixels spectral and spatial information. Thanks to the developed deep learning architecture the computed high level features were capable to outperformed the state-of-the-art in several experiments and hyperspectral datasets. The developed *DL-CNN* method was qualitatively and quantitatively compared with SVM-based classifiers (*RBF-SVM* and *Linear SVM*), deep learning models based on autoencoders (*DL-SAE*, (Chen et al., 2014)), and classification approaches based on Gaussian Process models (*GPMs*, (Ying Yang et al., 2015)), multilayer superpixel graph (*MSG-LBP*, (Zhan et al., 2015)) and adaptive sparse representations (*ASR*, (Li and Du, 2015)). The comprehensive experimental results and quantitative evaluation indicated the very high potentials of the developed approach.

Pavia University

Splitting ratio	5%	10%	20%	40%	80%
DL-SAE	90.55	92.67	94.32	96.17	97.58
DL-CNN	**97.13**	**97.88**	**98.73**	**99.51**	**99.88**
RBF-SVM	88.79	91.13	92.50	93.45	93.68
Linear SVM	81.04	82.24	82.38	82.75	83.07

Pavia Centre

Splitting ratio	5%	10%	20%	40%	80%
DL-SAE	99.30	99.46	99.56	99.60	99.73
DL-CNN	**99.64**	**99.74**	**99.85**	**99.95**	**99.98**
RBF-SVM	98.05	98.42	98.56	98.68	98.83
Linear SVM	97.22	97.29	97.34	97.39	97.48

KSC

Splitting ratio	5%	10%	20%	40%	80%
DL-SAE	41.79	70.88	73.36	88.61	92.65
DL-CNN	**64.07**	**75.13**	**86.41**	**91.47**	**96.25**
RBF-SVM	52.22	60.47	61.60	62.29	64.33
Linear SVM	49.66	59.65	63.70	68.83	72.96

Botswana

Splitting ratio	5%	10%	20%	40%	80%
DL-SAE	87.92	94.66	97.75	99.07	99.37
DL-CNN	**94.19**	**97.84**	**98.92**	**99.69**	**99.91**
RBF-SVM	75.13	87.78	90.80	94.35	96.61
Linear SVM	66.51	79.64	87.02	92.04	94.14

Indian Pines

Splitting ratio	5%	10%	20%	40%	80%
DL-SAE	70.62	75.96	81.08	87.13	92.87
DL-CNN	**81.76**	**87.34**	**93.19**	**96.82**	**99.20**
RBF-SVM	57.61	64.98	74.39	78.98	82.60
Linear SVM	63.60	71.45	74.87	77.06	79.33

Salinas

Splitting ratio	5%	10%	20%	40%	80%
DL-SAE	94.41	96.85	98.04	98.75	99.07
DL-CNN	**95.76**	**97.47**	**98.24**	**98.81**	**99.47**
RBF-SVM	91.61	92.35	92.81	93.23	93.74
Linear SVM	89.83	89.96	90.27	90.41	90.58

Table 11.5: Quantitative evaluation results for all datasets in terms of overall classification accuracy (%).

12

AFTERWORD

Throughout this dissertation, we presented application and/or scenario specific approaches for (i) detecting and identifying objects based exclusively on visual information, content and context and (ii) exploiting information and knowledge about objects in a scene as a basis for event understanding. All approaches follow the levels of understanding framework proposed by David Marr and they treat visual perception as an information processing problem.

Visual information processing and representation are both key aspects in every technical chapter of this thesis. As a representation considered a formal system for making explicit certain entities or types of information, while processing can be considered as the specification of how this representation takes place. The result of information processing and representation is a formal description of a given entity. Since each representation makes certain information explicit at the expense of information that is pushed into the background and may be quite hard to recover, it is inherently affects the performance and accuracy of vision algorithms. Therefore, the construction, selection and exploitation of highly descriptive features for representing visual information, according to the problem at hand, is a crucial step towards the development of *intelligent* vision-based systems.

However, the problem of intelligence – of how intelligence is created by the brain and of how to make intelligent machines – is one of the greatest problems in science, possibly the most fundamental of all. The realization that the brain is a computer is old. Alan Turing wrote about it and seeds of the idea can be found in centuries-old writings. However, we still do not understand the brain, of course this is not a surprise, and, thus, we are not able to develop machines able to conduct reasonable inference under a variety of real-world problems – *generalization* problem.

In this thesis, what we tried to do is to study a variety of vision problems at different levels of information organizations – from signal flows to logic reasoning. This way insights gained on higher levels helped us to ask reasonable questions and conduct valuable experiments at lower levels. We used realistic data, both synthetic, which are used to analyze noise sensitivity, and real-world data, in order to evaluate the performance of the proposed methodologies, in terms of objective criteria, and verify their validity.

Finally, under the perspective of generalization problem, computer vision is highly related to learning theory. It is important to study and understand the processing and representations of visual information, but it is also important to study and understand how an individual organism *learns* them. One could even argue that a description of the learning process and its a priori assumptions is deeper and more useful than a description of the details of what is actually learned. The problem of learning is at the core of the problem of intelligence and, thus, at the core of the problem of vision. For this reason, in this thesis, most of the presented methodologies conduct inference or make decisions by borrowing techniques from the learning theory. Not surprisingly, the language of statistical learning, including SVMs,

graphical models, neural nets, Bayesian inference, regularization, is permeating various areas of computer science.

To conclude, we need not only to understand the computational, algorithmic and hardware levels of vision problems, but also the way an individual could learn them. Only then may we be able to build intelligent machines that could learn to see.

BIBLIOGRAPHY

Achanta, R., Hemami, S., Estrada, F., and Susstrunk, S. (2009). Frequency-tuned salient region detection. In *IEEE Conference on Computer Vision and Pattern Recognition, 2009. CVPR 2009*, pages 1597–1604.

Achanta, R. and Susstrunk, S. (2010). Saliency detection using maximum symmetric surround. In *2010 17th IEEE International Conference on Image Processing (ICIP)*, pages 2653–2656.

Agarwal, S., Snavely, N., Simon, I., Seitz, S., and Szeliski, R. (2009). Building Rome in a day. In *2009 IEEE 12th International Conference on Computer Vision*, pages 72–79.

Agnew, D. J., Pearce, J., Pramod, G., Peatman, T., Watson, R., Beddington, J. R., and Pitcher, T. J. (2009). Estimating the Worldwide Extent of Illegal Fishing. *PLoS ONE*, 4(2).

Albrecht, T., Tan, T., West, G., Ly, T., and Moncrieff, S. (2011a). Vision-based attention in maritime environments. In *Communications and Signal Processing (ICICS) 2011 8th International Conference on Information*, pages 1–5.

Albrecht, T., West, G., Tan, T., and Ly, T. (2010). Multiple Views Tracking of Maritime Targets. In *2010 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*, pages 302–307.

Albrecht, T., West, G., Tan, T., and Ly, T. (2011b). Visual Maritime Attention Using Multiple Low-Level Features and Na #xoef;ve Bayes Classification. In *2011 International Conference on Digital Image Computing Techniques and Applications (DICTA)*, pages 243–249.

Alert, M. (2015). Medical alert - alarm and personal emergency response system by direct alert.

Alexe, B., Deselaers, T., and Ferrari, V. (2010). What is an object? In *2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 73–80.

Anderson, D., Luke, R. H., Keller, J. M., Skubic, M., Rantz, M., and Aud, M. (2009). Linguistic summarization of video for fall detection using voxel person and fuzzy logic. *Comput. Vis. Image Underst.*, 113(1):80–89.

Arampatzis, A., Zagoris, K., and Chatzichristofis, S. A. (2013). Dynamic two-stage image retrieval from large multimedia databases. *Information Processing & Management*, 49(1):274–285.

Auslander, B., Gupta, K. M., and Aha, D. W. (2011). A comparative evaluation of anomaly detection algorithms for maritime video surveillance. volume 8019, pages 801907–801907–14.

Auvinet, E., Multon, F., Saint-Arnaud, A., Rousseau, J., and Meunier, J. (2011). Fall detection with multiple cameras: An occlusion-resistant method based on 3-d silhouette vertical distribution. *IEEE Transactions on Information Technology in Biomedicine*, 15(2):290–300.

Bach, F. R. and Jordan, M. I. (2003). Learning Spectral Clustering. In *Advances in Neural Information Processing Systems 16*. MIT Press.

Ballard, D. H. and Brown, C. M. (1982). Computer vision. *Prenice-Hall, Englewood Cliffs, NJ*.

Barone, S., Paoli, A., and Razionale, A. (2012). 3d virtual reconstructions of artworks by a multiview scanning process. In *2012 18th International Conference on Virtual Systems and Multimedia (VSMM)*, pages 259–265.

Bastien, F., Lamblin, P., Pascanu, R., Bergstra, J., Goodfellow, I. J., Bergeron, A., Bouchard, N., and Bengio, Y. (2012). Theano: new features and speed improvements. Deep Learning and Unsupervised Feature Learning NIPS 2012 Workshop.

Bay, H., Ess, A., Tuytelaars, T., and Van Gool, L. (2008). Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359.

Bay, H., Tuytelaars, T., and Van Gool, L. (2006). Surf: Speeded up robust features. In *Computer vision–ECCV 2006*, pages 404–417. Springer.

Belkin, M. and Niyogi, P. (2003). Using Manifold Stucture for Partially Labeled Classification. In Becker, S., Thrun, S., and Obermayer, K., editors, *Advances in Neural Information Processing Systems 15*, pages 953–960. MIT Press.

Belongie, S., Malik, J., and Puzicha, J. (2002). Shape matching and object recognition using shape contexts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(4):509–522.

Bengio, Y., Lamblin, P., Popovici, D., Larochelle, H., et al. (2007). Greedy layer-wise training of deep networks. *Advances in neural information processing systems*, 19:153.

Bergstra, J., Breuleux, O., Bastien, F., Lamblin, P., Pascanu, R., Desjardins, G., Turian, J., Warde-Farley, D., and Bengio, Y. (2010). Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*. Oral Presentation.

Bevilacqua, A., Gherardi, A., and Carozza, L. (2008). Automatic perspective camera calibration based on an incomplete set of chessboard markers. In *Sixth Indian Conference on Computer Vision, Graphics Image Processing, 2008. ICVGIP '08*, pages 126–133.

Bianchi, F., Redmond, S., Narayanan, M., Cerutti, S., and Lovell, N. (2010). Barometric pressure and triaxial accelerometry-based falls event detection. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 18(6):619–627.

Bishop, C. (2007). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer.

Black, M. J. and Yacoob, Y. (1997). Recognizing facial expressions in image sequences using local parameterized models of image motion. *International Journal of Computer Vision*, 25(1):23–48.

Boden, M. (2006). *Mind As Machine: A History of Cognitive Science Two-Volume Set*. Oxford University Press, Oxford : New York.

Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer.

Bouwmans, T., El Baf, F., Vachon, B., et al. (2010). Statistical background modeling for foreground detection: A survey. *Handbook of Pattern Recognition and Computer Vision*, 4(2):181–189.

Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press.

Brutzer, S., Hoferlin, B., and Heidemann, G. (2011). Evaluation of background subtraction techniques for video surveillance. In *2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1937–1944.

Bunsch, E., Guzowska, A., and Sitnik, R. (2012). 3d scanning documentation of two different objects - The King's Chinese Cabinet in Wilanow Palace Museum and a Roman gravestone from archeological excavations in Moesia Inferior as a part of multidisciplinary research. In *2012 18th International Conference on Virtual Systems and Multimedia (VSMM)*, pages 633–636.

Burt, P. J. and Adelson, E. H. (1983). The laplacian pyramid as a compact image code. *Communications, IEEE Transactions on*, 31(4):532–540.

Calonder, M., Lepetit, V., Strecha, C., and Fua, P. (2010). BRIEF: Binary Robust Independent Elementary Features. In Daniilidis, K., Maragos, P., and Paragios, N., editors, *Computer Vision - ECCV 2010*, number 6314 in Lecture Notes in Computer Science, pages 778–792. Springer Berlin Heidelberg.

Camps-Valls, G. and Bruzzone, L. (2005). Kernel-based methods for hyperspectral image classification. *Geoscience and Remote Sensing, IEEE Transactions on*, 43(6):1351–1362.

Camps-Valls, G. and Bruzzone, L. (2009). *Kernel methods for remote sensing data analysis*. John Wiley & Sons.

Camps-Valls, G., Tuia, D., Bruzzone, L., and Atli Benediktsson, J. (2014). Advances in hyperspectral image classification: Earth monitoring with statistical learning methods. *Signal Processing Magazine, IEEE*, 31(1):45–54.

Cavanagh, P. (1992). Multiple analyses of orientation in the visual system.

Cayton, L. (2006). Algorithms for manifold learning. Tech. Rep. CS2008-0923, University of California, San Diego.

Chang, C.-I. (2013). *Hyperspectral data processing: algorithm design and analysis*. John Wiley & Sons.

Chatzis, S. and Varvarigou, T. (2009). Factor analysis latent subspace modeling and robust fuzzy clustering using -distributions. *IEEE Transactions on Fuzzy Systems*, 17(3):505–517.

Chauvin, Y. and Rumelhart, D. E. (1995). *Backpropagation: theory, architectures, and applications*. Psychology Press.

Chen, Y., Lin, Z., Zhao, X., Wang, G., and Gu, Y. (2014). Deep learning-based classification of hyperspectral data. *Selected Topics in Applied Earth Observations and Remote Sensing, IEEE Journal of*, 7(6):2094–2107.

Cheung, S.-C. S. and Kamath, C. (2005). Robust background subtraction with foreground validation for urban traffic video. *EURASIP Journal on Advances in Signal Processing*, 2005(14):726261.

Chum, O., Philbin, J., Sivic, J., Isard, M., and Zisserman, A. (2007). Total Recall: Automatic Query Expansion with a Generative Feature Model for Object Retrieval. In *IEEE 11th International Conference on Computer Vision, 2007. ICCV 2007*, pages 1–8.

Cortes, C. and Vapnik, V. (1995a). Support-vector networks. *Machine learning*, 20(3):273–297.

Cortes, C. and Vapnik, V. (1995b). Support-vector networks. *Machine Learning*, 20(3):273–297.

Cox, M. A. A. and Cox, T. F. (2008). Multidimensional Scaling. In *Handbook of Data Visualization*, Springer Handbooks Comp.Statistics, pages 315–347. Springer Berlin Heidelberg.

Crevier, D. (1993). *AI: The Tumultuous History of the Search for Artificial Intelligence*. Basic Books, Inc., New York, NY, USA.

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

Cyganek, B. (2007). *An Introduction to 3D Computer Vision Techniques and Algorithms*. John Wiley & Sons.

Dai, C., Zheng, Y., and Li, X. (2007). Pedestrian detection and tracking in infrared imagery using shape and appearance. *Computer Vision and Image Understanding*, 106(2-3):288–299.

Dasgupta, S. and Hsu, D. (2007). On-line estimation with the multivariate gaussian distribution. In *Proceedings of the 20th Annual Conference on Learning Theory*, COLT'07, pages 278–292, Berlin, Heidelberg. Springer-Verlag.

Davis, J. (2001a). Hierarchical motion history images for recognizing human motion. In *Detection and Recognition of Events in Video, 2001. Proceedings. IEEE Workshop on*, pages 39–46.

Davis, J. and Sharma, V. (2005). Fusion-based background-subtraction using contour saliency. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops, 2005. CVPR Workshops*, pages 11–11.

Davis, J. W. (2001b). Hierarchical motion history images for recognizing human motion. In *Detection and Recognition of Events in Video, 2001. Proceedings. IEEE Workshop on*, pages 39–46. IEEE.

Davis, J. W. and Sharma, V. (2004). Robust background-subtraction for person detection in thermal imagery. In *Proceedings of the 2004 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'04) Volume 8 - Volume 08*, CVPRW '04, pages 128–, Washington, DC, USA. IEEE Computer Society.

Davis, J. W. and Sharma, V. (2007). Background-subtraction in thermal imagery using contour saliency. *International Journal of Computer Vision*, 71(2):161–181.

Debard, G., Karsmakers, P., Deschodt, M., Vlaeyen, E., Dejaeger, E., Milisen, K., GoedemÃ©, T., Vanrumste, B., and Tuytelaars, T. (2012). Camera-based fall detection on real world data. In Dellaert, F., Frahm, J.-M., Pollefeys, M., Leal-TaixÃ©, L., and Rosenhahn, B., editors, *Outdoor and Large-Scale Real-World Scene Analysis*, number 7474 in Lecture Notes in Computer Science, pages 356–375. Springer Berlin Heidelberg.

Diraco, G., Leone, A., and Siciliano, P. (2010). An active vision system for fall detection and posture recognition in elderly healthcare. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2010*, pages 1536–1541.

Doulamis, A. and Doulamis, N. (2004). Generalized nonlinear relevance feedback for interactive content-based retrieval and organization. *IEEE Transactions on Circuits and Systems for Video Technology*, 14(5):656–671.

Doulamis, A. and Makantasis, K. (2011). Iterative scene learning in visually guided persons falls detection. In *European Signal processing conference*, pages 779–783, Barcelona-Spain.

Doulamis, A. D., Doulamis, N. D., and Kollias, S. D. (2000). A fuzzy video content representation for video summarization and content-based retrieval. *Signal Processing*, 80(6):1049–1067.

Doulamis, N. (2010). Iterative motion estimation constrained by time and shape for detecting persons falls. In *Proceedings of the 3rd International Conference on PErvasive Technologies Related to Assistive Environments*, PETRA '10, pages 62:1–62:8, New York, NY, USA. ACM.

Doulamis, N. and Doulamis, A. (2006). Evaluation of relevance feedback schemes in content-based in retrieval systems. *Signal Processing: Image Communication*, 21(4):334–357.

Doulamis, N. and Doulamis, A. (2012). Fast and Adaptive Deep Fusion Learning for Detecting Visual Objects. In Fusiello, A., Murino, V., and Cucchiara, R., editors, *Computer Vision - ECCV 2012. Workshops and Demonstrations*, number 7585 in Lecture Notes in Computer Science, pages 345–354. Springer Berlin Heidelberg.

Doulamis, N., Doulamis, A., and Varvarigou, T. (2003). Adaptive algorithms for interactive multimedia. *IEEE MultiMedia*, 10(4):38–47.

Dubey, R., Ni, B., and Moulin, P. (2012). A depth camera based fall recognition system for the elderly. In Campilho, A. and Kamel, M., editors, *Image Analysis and Recognition*, number 7325 in Lecture Notes in Computer Science, pages 106–113. Springer Berlin Heidelberg.

Efros, A., Berg, A., Mori, G., and Malik, J. (2003). Recognizing action at a distance. In *Ninth IEEE International Conference on Computer Vision, 2003. Proceedings*, pages 726–733 vol.2.

El Baf, F., Bouwmans, T., and Vachon, B. (2009). Fuzzy statistical modeling of dynamic backgrounds for moving object detection in infrared videos. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, 2009. CVPR Workshops 2009*, pages 60–65.

Elgammal, A., Harwood, D., and Davis, L. (2000). Non-parametric model for background subtraction. In Vernon, D., editor, *Computer Vision - ECCV 2000*, number 1843 in Lecture Notes in Computer Science, pages 751–767. Springer Berlin Heidelberg.

Elguebaly, T. and Bouguila, N. (2013). Finite asymmetric generalized gaussian mixture models learning for infrared object detection. *Computer Vision and Image Understanding*, 117(12):1659–1671.

Ester, M., Kriegel, H.-p., S, J., and Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. pages 226–231. AAAI Press.

Fallwatch (2009). FallWatch project - a wearable miniaturized fall detection system for the elderly.

Fan, K. (1951). Maximum Properties and Inequalities for the Eigenvalues of Completely Continuous Operators. *Proceedings of the National Academy of Sciences of the United States of America*, 37(11):760–766.

Fischer, Y. and Bauer, A. (2010). Object-oriented sensor data fusion for wide maritime surveillance. In *Waterside Security Conference (WSS), 2010 International*, pages 1–6.

Foroughi, H., Aski, B., and Pourreza, H. (2008a). Intelligent video surveillance for monitoring fall detection of elderly in home environments. In *11th International Conference on Computer and Information Technology, 2008. ICCIT 2008*, pages 219–224.

Foroughi, H., Rezvanian, A., and Paziraee, A. (2008b). Robust fall detection using human shape and multi-class support vector machine. In *Sixth Indian Conference on Computer Vision, Graphics Image Processing, 2008. ICVGIP '08*, pages 413–420.

Fu, Z., Culurciello, E., Lichtsteiner, P., and Delbruck, T. (2008). Fall detection using an address-event temporal contrast vision sensor. In *IEEE International Symposium on Circuits and Systems, 2008. ISCAS 2008*, pages 424–427.

Gade, R., Jorgensen, A., and Moeslund, T. (2013). Long-term occupancy analysis using graph-based optimisation in thermal imagery. In *2013 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3698–3705.

Grammatikopoulos, L., Karras, G., and Petsa, E. (2007). An automatic approach for camera calibration from vanishing points. *ISPRS Journal of Photogrammetry and Remote Sensing*, pages 64–76.

Grassi, M., Lombardi, A., Rescio, G., Ferri, M., Malcovati, P., Leone, A., Diraco, G., Siciliano, P., Malfatti, M., and Gonzo, L. (2010). An integrated system for people fall-detection with data fusion capabilities based on 3d ToF camera and wireless accelerometer. In *2010 IEEE Sensors*, pages 1016–1019.

Haines, T. and Xiang, T. (2014). Background subtraction with DirichletProcess mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(4):670–683.

Halko, N., Martinsson, P., and Tropp, J. (2009). Finding structure with randomness: Stochastic algorithms for constructing approximate matrix decompositions, 2009. *URL http://arxiv. org/abs/0909*, 4061.

Halkos, D., Doulamis, N., and Doulamis, A. (2009). A secure framework exploiting content guided and automated algorithms for real time video searching. *Multimedia Tools and Applications*, 42(3):343–375.

Harris, C. and Stephens, M. (1988). A combined corner and edge detector. In *Alvey vision conference*, volume 15, page 50. Citeseer.

Hazelhoff, L., Han, J., and With, P. H. (2008). Video-based fall detection in the home using principal component analysis. In *Proceedings of the 10th International Conference on Advanced Concepts for Intelligent Vision Systems*, ACIVS '08, pages 298–309, Berlin, Heidelberg. Springer-Verlag.

Herrero, S. and Bescos, J. (2009). Background subtraction techniques: Systematic evaluation and comparative analysis. In *Proceedings of the 11th International Conference on Advanced Concepts for Intelligent Vision Systems*, ACIVS '09, pages 33–42, Berlin, Heidelberg. Springer-Verlag.

Hinton, G. E., Osindero, S., and Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554.

Hinton, G. E. and Roweis, S. T. (2002). Stochastic neighbor embedding. In Becker, S., Thrun, S., and Obermayer, K., editors, *Advances in NIPS 15*, pages 833–840.

Hinton, G. E. and Salakhutdinov, R. R. (2006a). Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507.

Hinton, G. E. and Salakhutdinov, R. R. (2006b). Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507.

Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, 24(6):417.

Hyvärinen, A., Karhunen, J., and Oja, E. (2004). *Independent component analysis*, volume 46. John Wiley & Sons.

Janssens, J., Huszar, F., Postma, E., and van den Herik, J. (2012). Stochastic outlier selection. Technical Report TiCC TR 2012-001, Tilburg University, Netherlands.

Jhuang, H., Serre, T., Wolf, L., and Poggio, T. (2007). A biologically inspired system for action recognition. In *IEEE 11th International Conference on Computer Vision, 2007. ICCV 2007*, pages 1–8.

Ji, S., Xu, W., Yang, M., and Yu, K. (2013). 3d convolutional neural networks for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):221–231.

Jolliffe, I. (2002). *Principal component analysis*. Wiley Online Library.

Juan, L. and Gwun, O. (2009). A comparison of sift, pca-sift and surf. *International Journal of Image Processing (IJIP)*, 3(4):143–152.

Jungling, K. and Arens, M. (2009). Feature based person detection beyond the visible spectrum. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, 2009. CVPR Workshops 2009*, pages 30–37.

Kaimakis, P. and Tsapatsoulis, N. (2013). Background Modeling Methods for Visual Detection of Maritime Targets. In *Proceedings of the 4th ACM/IEEE International Workshop on Analysis and Retrieval of Tracked Events and Motion in Imagery Stream*, ARTEMIS '13, pages 67–76, New York, NY, USA. ACM.

Karaszewski, M., Sitnik, R., and Bunsch, E. (2012). On-line, collision-free positioning of a scanner during fully automated three-dimensional measurement of cultural heritage objects. *Robotics and Autonomous Systems*, 60(9):1205–1219.

Ke, Y. and Sukthankar, R. (2004). Pca-sift: A more distinctive representation for local image descriptors. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 2, pages II–506. IEEE.

Kekre, D. H. B., Sarode, T. K., Thepade, S. D., and Vaishali, V. (2011). Improved texture feature based image retrieval using kekre's fast codebook generation algorithm. In Pise, S. J., editor, *Thinkquest~2010*, pages 143–149. Springer India.

Kirby, M. and Sirovich, L. (1990). Application of the karhunen-loeve procedure for the characterization of human faces. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 12(1):103–108.

Kolmogorov, A. N. (1963). On the representation of continuous functions of many variables by superposition of continuous functions of one variable and addition. *Amer. Math. Soc. Transl*, 28:55–59.

Kosmopoulos, D. I., Doulamis, A., Makris, A., Doulamis, N., Chatzis, S., and Middleton, S. E. (2009). Vision-based production of personalized video. *Signal Processing: Image Communication*, 24(3):158–176.

Kosmopoulos, D. I., Doulamis, N. D., and Voulodimos, A. S. (2012). Bayesian filter based behavior recognition in workflows allowing for user feedback. *Computer Vision and Image Understanding*, 116(3):422–434.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

Laptev, I. and Perez, P. (2007). Retrieving actions in movies. In *IEEE 11th International Conference on Computer Vision, 2007. ICCV 2007*, pages 1–8.

Latecki, L., Miezianko, R., and Pokrajac, D. (2005). Tracking motion objects in infrared videos. In *IEEE Conference on Advanced Video and Signal Based Surveillance, 2005. AVSS 2005*, pages 99–104.

Le, T. and Pan, R. (2009). Accelerometer-based sensor network for fall detection. In *IEEE Biomedical Circuits and Systems Conference, 2009. BioCAS 2009*, pages 265–268.

Le Roux, N. and Bengio, Y. (2010). Deep belief networks are compact universal approximators. *Neural computation*, 22(8):2192–2207.

Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Lee, D. D. and Seung, H. S. (1999). Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791.

Lee, H., Grosse, R., Ranganath, R., and Ng, A. Y. (2009). Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 609–616, New York, NY, USA. ACM.

Lei, P.-R. (2013). Exploring trajectory behavior model for anomaly detection in maritime moving objects. In *2013 IEEE International Conference on Intelligence and Security Informatics (ISI)*, pages 271–271.

Li, W. and Du, Q. (2015). Adaptive sparse representation for hyperspectral image classification. In *IEEE International Geoscience and Remote Sensing Symposium (IGARSS 2015)*.

Lindeberg, T. (1998). Feature detection with automatic scale selection. *International journal of computer vision*, 30(2):79–116.

Liu, T., Yuan, Z., Sun, J., Wang, J., Zheng, N., Tang, X., and Shum, H.-Y. (2011). Learning to Detect a Salient Object. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(2):353–367.

Liu, W., He, J., and Chang, S.-f. *Large Graph Construction for Scalable Semi-Supervised Learning*.

Liu, Y., Yao, H., Gao, W., Chen, X., and Zhao, D. (2007). Nonparametric background generation. *J. Vis. Comun. Image Represent.*, 18(3):253–263.

Livingstone, M. (2002). *Vision and art: the biology of seeing*. Harry N. Abrams.

Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110.

Lucas, B. D. and Kanade, T. (1981). An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'81, pages 674–679, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Lv, Q., Josephson, W., Wang, Z., Charikar, M., and Li, K. (2007). Multi-probe LSH: Efficient Indexing for High-dimensional Similarity Search. In *Proceedings of the 33rd International Conference on Very Large Data Bases*, VLDB '07, pages 950–961, Vienna, Austria. VLDB Endowment.

Makantasis, K., Doulamis, A., and Doulamis, N. (2013). Vision-based maritime surveillance system using fused visual attention maps and online adaptable tracker. In *2013 14th International Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS)*, pages 1–4.

Makantasis, K., Doulamis, A., and Matsatsinis, N. (2012). Student-t background modeling for persons' fall detection through visual cues. In *2012 13th International Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS)*, pages 1–4.

Makantasis, K., Karantzalos, K., Doulamis, A., and Doulamis, N. (2015). Deep Supervised Learning for Hyperspectral Data Classification through Convolutional Neural Networks. In *IEEE International Geoscience and Remote Sensing Symposium (IGARSS 2015)*.

Maresca, S., Greco, M., Gini, F., Grasso, R., Coraluppi, S., and Horstmann, J. (2010). Vessel detection and classification: An integrated maritime surveillance system in the Tyrrhenian sea. In *2010 2nd International Workshop on Cognitive Information Processing (CIP)*, pages 40–45.

Marr, D., Poggio, T. A., and Ullman, S. (2010). *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. The MIT Press, Cambridge, Mass.

Mastorakis, G. and Makris, D. (2012). Fall detection system using kinect's infrared sensor. *Journal of Real-Time Image Processing*, 9(4):635–646.

Matas, J., Chum, O., Urban, M., and Pajdla, T. (2004). Robust wide-baseline stereo from maximally stable extremal regions. *Image and vision computing*, 22(10):761–767.

McFarlane, N. J. B. and Schofield, C. P. (1995). Segmentation and tracking of piglets in images. *Machine Vision and Applications*, 8(3):187–193.

McIlhagga, W. (2010). The Canny Edge Detector Revisited. *International Journal of Computer Vision*, 91(3):251–261.

Messelodi, S., Modena, C. M., Segata, N., and Zanin, M. (2005). A kalman filter based background updating algorithm robust to sharp illumination changes. In Roli, F. and Vitulano, S., editors, *Image Analysis and Processing – ICIAP 2005*, number 3617 in Lecture Notes in Computer Science, pages 163–170. Springer Berlin Heidelberg.

Mikolajczyk, K. and Schmid, C. (2002). An affine invariant interest point detector. In *Computer Vision - ECCV 2002*, pages 128–142. Springer.

Mikolajczyk, K. and Schmid, C. (2005). A performance evaluation of local descriptors. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(10):1615–1630.

Min, R. and Cheng, H. (2009). Effective image retrieval using dominant color descriptor and fuzzy support vector machine. *Pattern Recognition*, 42(1):147–157.

Mnih, V. and Hinton, G. E. (2012). Learning to label aerial images from noisy data. In *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, pages 567–574.

Murthy, V. S. V. S., Kumar, S., and Rao, P. S. (2010). *Content Based Image Retrieval using Hierarchical and K-Means Clustering Techniques*.

Nadler, B., Srebro, N., and Zhou, X. (2009). Statistical analysis of semi-supervised learning: The limit of infinite unlabelled data. In *Advances in Neural Information Processing Systems*, pages 1330–1338.

Nilsson, M., van Laere, J., Ziemke, T., and Edlund, J. (2008). Extracting rules from expert operators to support situation awareness in maritime surveillance. In *2008 11th International Conference on Information Fusion*, pages 1–8.

Noury, N., Fleury, A., Rumeau, P., Bourke, A., Laighin, G., Rialle, V., and Lundy, J. (2007). Fall detection - principles and methods. In *29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, 2007. EMBS 2007*, pages 1663–1666.

Ntalianis, K. S., Doulamis, A. D., Tsapatsoulis, N., and Doulamis, N. (2010). Human action annotation, modeling and analysis based on implicit user interaction. *Multimedia Tools and Applications*, 50(1):199–225.

Nyan, M. N., Tay, F. E. H., and Murugasu, E. (2008). A wearable system for pre-impact fall detection. *Journal of Biomechanics*, 41(16):3475–3481.

Ojala, T., Pietikäinen, M., and Harwood, D. (1996). A comparative study of texture measures with classification based on featured distributions. *Pattern recognition*, 29(1):51–59.

Onuoha, F. (2009). Sea piracy and maritime security in the Horn of Africa: The Somali coast and Gulf of Aden in perspective. *African Security Review*, 18(3):31–44.

Orr, G. B. and Müller, K.-R. (2003). *Neural networks: tricks of the trade*. Springer.

Palmer, S. E. (1999). *Vision Science: Photons to Phenomenology*. The MIT Press, Cambridge, Mass, 1 edition edition.

Papadakis, N., Doulamis, A., Litke, A., Doulamis, N., Skoutas, D., and Varvarigou, T. (2008). MI-MERCURY: A mobile agent architecture for ubiquitous retrieval and delivery of multimedia information. *Multimedia Tools and Applications*, 38(1):147–184.

Papadopoulos, S., Zigkolis, C., Kompatsiaris, Y., and Vakali, A. (2010). Cluster-based Landmark and Event Detection on Tagged Photo Collections. *IEEE Multimedia*.

Person, K. (1901). On lines and planes of closest fit to system of points in space. philiosophical magazine, 2, 559-572.

Pham, Q.-C., Gond, L., Begard, J., Allezard, N., and Sayd, P. (2007). Real-time posture analysis in a crowd using thermal imaging. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE.

Phansalkar, V. and Sastry, P. (1994). Analysis of the back-propagation algorithm with momentum. *IEEE Transactions on Neural Networks*, 5(3):505–506.

Philbin, J., Chum, O., Isard, M., Sivic, J., and Zisserman, A. (2007). Object retrieval with large vocabularies and fast spatial matching. In *IEEE Conference on Computer Vision and Pattern Recognition, 2007. CVPR '07*, pages 1–8.

Polana, R. and Nelson, R. (1994). Detecting activities. *Journal of Visual Communication and Image Representation*, 5(2):172–180.

Porikli, F. (2006). Achieving real-time object detection and tracking under extreme conditions. *Journal of Real-Time Image Processing*, 1(1):33–40.

Protopapadakis, E., Doulamis, A., and Doulamis, N. (2013). Tapped delay multiclass support vector machines for industrial workflow recognition. In *2013 14th International Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS)*, pages 1–4.

Qian, H., Mao, Y., Xiang, W., and Wang, Z. (2008). Home environment fall detection system based on a cascaded multi-SVM classifier. In *10th International Conference on Control, Automation, Robotics and Vision, 2008. ICARCV 2008*, pages 1567–1572.

Rodriguez Sullivan, M. D. and Shah, M. (2008). Visual surveillance in maritime port facilities. volume 6978, pages 697811–697811–8.

Rojas, R. (2013). *Neural networks: a systematic introduction*. Springer Science & Business Media.

Rosin, P. L. (1999). Measuring Corner Properties. *Computer Vision and Image Understanding*, 73(2):291–307.

Rosten, E. and Drummond, T. (2006). Machine Learning for High-Speed Corner Detection. In Leonardis, A., Bischof, H., and Pinz, A., editors, *Computer Vision - ECCV 2006*, number 3951 in Lecture Notes in Computer Science, pages 430–443. Springer Berlin Heidelberg.

Rother, C. (2002). A new approach to vanishing point detection in architectural environments. *Image and Vision Computing*, 20(9-10):647–655.

Rougier, C., Auvinet, E., Rousseau, J., Mignotte, M., and Meunier, J. (2011a). Fall detection from depth map video sequences. In *Proceedings of the 9th International Conference on Toward Useful Services for Elderly and People with Disabilities: Smart Homes and Health Telematics*, ICOST'11, pages 121–128, Berlin, Heidelberg. Springer-Verlag.

Rougier, C., Meunier, J., St-Arnaud, A., and Rousseau, J. (2011b). Robust video surveillance for fall detection based on human shape deformation. *IEEE Transactions on Circuits and Systems for Video Technology*, 21(5):611–622.

Rublee, E., Rabaud, V., Konolige, K., and Bradski, G. (2011). ORB: An efficient alternative to SIFT or SURF. In *2011 IEEE International Conference on Computer Vision (ICCV)*, pages 2564–2571.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1985). Learning internal representations by error propagation. Technical report, DTIC Document.

Runge, J. W. (1993). The cost of injury. *Emergency Medicine Clinics of North America*, 11(1):241–253.

Salakhutdinov, R. and Hinton, G. (2009a). Deep boltzmann machines. In *In: Proc. of the Int. Conf. on ArtificialIntelligence andStatistics (AISTATS2009).pp. 448-455*.

Salakhutdinov, R. and Hinton, G. E. (2009b). Deep boltzmann machines. In *International Conference on Artificial Intelligence and Statistics*, pages 448–455.

Salakhutdinov, R., Mnih, A., and Hinton, G. (2007). Restricted boltzmann machines for collaborative filtering. In *Proceedings of the 24th international conference on Machine learning*, pages 791–798. ACM.

Sanderson, J. (1999). Target identification in a complex maritime scene. In *1999/103), IEE Colloquium on Motion Analysis and Tracking (Ref. No*, pages 15/1–15/4.

Satopaa, V., Albrecht, J., Irwin, D., and Raghavan, B. (2011). Finding a "Kneedle" in a Haystack: Detecting Knee Points in System Behavior. In *Proceedings of the 2011 31st International Conference on Distributed Computing Systems Workshops*, ICDCSW '11, pages 166–171, Washington, DC, USA. IEEE Computer Society.

Schindler, K. and Van Gool, L. (2008). Action snippets: How many frames does human action recognition require? In *IEEE Conference on Computer Vision and Pattern Recognition, 2008. CVPR 2008*, pages 1–8.

Schmid, C., Mohr, R., and Bauckhage, C. (2000). Evaluation of interest point detectors. *International Journal of computer vision*, 37(2):151–172.

Shi, J. and Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905.

Shi, J. and Tomasi, C. (1994). Good features to track. In *, 1994 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1994. Proceedings CVPR '94*, pages 593–600.

Shin, H.-C., Orton, M. R., Collins, D. J., Doran, S. J., and Leach, M. O. (2013). Stacked autoencoders for unsupervised feature learning and multiple organ detection in a pilot study using 4d patient data. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(8):1930–1943.

Shrestha, L. and Heisler, E. (2011). The changing demographic profile of the united states. *Federal Publications*.

Simard, P. Y., Steinkraus, D., and Platt, J. C. (2003). Best practices for convolutional neural networks applied to visual document analysis. In *null*, page 958. IEEE.

Simon, I., Snavely, N., and Seitz, S. (2007). Scene Summarization for Online Image Collections. In *IEEE 11th International Conference on Computer Vision, 2007. ICCV 2007*, pages 1–8.

Sitnik, R. and Karaszewski, M. (2010). Automated Processing of Data from 3d Scanning of Cultural Heritage Objects. In Ioannides, M., Fellner, D., Georgopoulos, A., and Hadjimitsis, D. G., editors, *Digital Heritage*, number 6436 in Lecture Notes in Computer Science, pages 28–41. Springer Berlin Heidelberg.

Sobottka, K. and Pitas, I. (1996). Face localization and facial feature extraction based on shape and color information. In *, International Conference on Image Processing, 1996. Proceedings*, volume 3, pages 483–486 vol.3.

Socek, D., Culibrk, D., Marques, O., Kalva, H., and Furht, B. (2005). A Hybrid Color-Based Foreground Object Detection Method for Automated Marine Surveillance. In Blanc-Talon, J., Philips, W., Popescu, D., and Scheunders, P., editors, *Advanced Concepts for Intelligent Vision Systems*, number 3708 in Lecture Notes in Computer Science, pages 340–347. Springer Berlin Heidelberg.

Spiliotis, I. and Mertzios, B. (1998). Real-time computation of two-dimensional moments on binary images using image block representation. *IEEE Transactions on Image Processing*, 7(11):1609–1615.

Stanslas, P. T. (2010). Transborder Human Trafficking in Malaysian Waters: Addressing the Root Causes. *Journal of Maritime Law and Commerce*, 41:595.

Stauffer, C. and Grimson, W. (1999). Adaptive background mixture models for real-time tracking. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 2, pages –252 Vol. 2.

Sutskever, I. and Hinton, G. E. (2008). Deep, narrow sigmoid belief networks are universal approximators. *Neural Computation*, 20(11):2629–2636.

Szeliski, R. (2010). *Computer Vision: Algorithms and Applications*. Springer-Verlag New York, Inc., New York, NY, USA, 1st edition.

Szpak, Z. L. and Tapamo, J. R. (2011). Maritime surveillance: Tracking ships inside a dynamic background using a fast level-set. *Expert Systems with Applications*, 38(6):6669–6680.

Thome, N., Miguet, S., and Ambellouis, S. (2008). A real-time, multiview fall detection system: A LHMM-based approach. *IEEE Transactions on Circuits and Systems for Video Technology*, 18(11):1522–1532.

Thormaehlen, T., Broszio, H., and Wassermann, I. (2003). Robust line-based calibration of lens distortion from a single view. In *ftp://ftp.tnt.uni-hannover.de/pub/papers/2003/MIRA03-TTHBIW.pdf*. Thorsten Thormaehlen, Hellward Broszio and Ingolf Wassermann.

Toyama, K., Krumm, J., Brumitt, B., and Meyers, B. (1999). Wallflower: principles and practice of background maintenance. In *The Proceedings of the Seventh IEEE International Conference on Computer Vision, 1999*, volume 1, pages 255–261 vol.1.

Turk, M. and Pentland, A. (1991). Eigenfaces for recognition. *Journal of cognitive neuroscience*, 3(1):71–86.

Tuzel, O., Porikli, F., and Meer, P. (2007). Human detection via classification on riemannian manifolds. In *IEEE Conference on Computer Vision and Pattern Recognition, 2007. CVPR '07*, pages 1–8.

Tuzel, O., Porikli, F., and Meer, P. (2008). Pedestrian detection via classification on riemannian manifolds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(10):1713–1727.

Van Ooyen, A. and Nienhuis, B. (1992). Improving the convergence of the back-propagation algorithm. *Neural Networks*, 5(3):465–471.

Vandecasteele, A., Devillers, R., and Napoli, A. (2013). A semi-supervised learning framework based on spatio-temporal semantic events for maritime anomaly detection and behavior analysis. page 4 pages.

Vapnik, V. N. (1995). *The nature of statistical learning theory*. Springer-Verlag New York, Inc.

Vapnik, V. N. (1998). *Statistical learning theory*, volume 1. Wiley New York.

Veres, G., Grabner, H., Middleton, L., and Gool, L. V. (2011). Automatic workflow monitoring in industrial environments. In Kimmel, R., Klette, R., and Sugimoto, A., editors, *Computer Vision – ACCV 2010*, number 6492 in Lecture Notes in Computer Science, pages 200–213. Springer Berlin Heidelberg.

Vincent, P., Larochelle, H., Bengio, Y., and Manzagol, P.-A. (2008). Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103. ACM.

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., and Manzagol, P.-A. (2010). Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *The Journal of Machine Learning Research*, 11:3371–3408.

Voulodimos, A., Kosmopoulos, D., Vasileiou, G., Sardis, E., Doulamis, A., Anagnostopoulos, V., Lalos, C., and Varvarigou, T. (2011). A dataset for workflow recognition in industrial scenes. In *Image Processing (ICIP), 2011 18th IEEE International Conference on*, pages 3249–3252. IEEE.

Wang, S., Yang, J., Chen, N., Chen, X., and Zhang, Q. (2005). Human activity recognition with user-free accelerometers in the sensor networks. In *International Conference on Neural Networks and Brain, 2005. ICNN B '05*, volume 2, pages 1212–1217.

Wang, W., Zhang, J., and Shen, C. (2010). Improved human detection and classification in thermal images. In *2010 17th IEEE International Conference on Image Processing (ICIP)*, pages 2313–2316.

Wijnhoven, R., Rens, K., Jaspers, E., and With, P. (2010). Online learning for ship detection in maritime surveillance.

Winter, M. E. (1999). N-FINDR: an algorithm for fast autonomous spectral end-member determination in hyperspectral data. volume 3753, pages 266–275.

Wren, C., Azarbayejani, A., Darrell, T., and Pentland, A. (1997). Pfinder: real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):780–785.

Wu, C., Agarwal, S., Curless, B., and Seitz, S. (2011a). Multicore bundle adjustment. In *2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3057–3064.

Wu, C., Agarwal, S., Curless, B., and Seitz, S. (2012). Schematic surface reconstruction. In *2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1498–1505.

Wu, C., Frahm, J.-M., and Pollefeys, M. (2011b). Repetition-based dense single-view reconstruction. In *2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3113–3120.

Xiang, T. and Gong, S. (2006). Beyond tracking: Modelling activity and understanding behaviour. *International Journal of Computer Vision*, 67(1):21–51.

Yasri, I., Hamid, N., and Yap, V. (2008). Performance analysis of FPGA based Sobel edge detection operator. In *International Conference on Electronic Design, 2008. ICED 2008*, pages 1–4.

Ying Yang, M., Liao, W., RosenHahn, B., and Zhang, Z. (2015). Hyperspectral image classification using gaussian process models. In *IEEE International Geoscience and Remote Sensing Symposium (IGARSS 2015)*.

Yu, S. and Shi, J. (2003). Multiclass spectral clustering. In *Ninth IEEE International Conference on Computer Vision, 2003. Proceedings*, pages 313–319 vol.1.

Zelnik-Manor, L. and Irani, M. (2001). Event-based analysis of video. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 2, pages II–123. IEEE.

Zemmari, R., Daun, M., Feldmann, M., and Nickel, U. (2013). Maritime surveillance with GSM passive radar: Detection and tracking of small agile targets. In *Radar Symposium (IRS), 2013 14th International*, volume 1, pages 245–251.

Zhan, T., Xu, Y., Sun, L., Wu, Z., and Zhan, Y. (2015). Hyperspectral image classification using multilayer superpixel graph and loopy belief propagation. In *IEEE International Geoscience and Remote Sensing Symposium (IGARSS 2015)*.

Zheng, J., Wang, Y., Nihan, N., and Hallenbeck, M. (2006). Extracting roadway background image: Mode-based approach. *Transportation Research Record: Journal of the Transportation Research Board*, 1944:82–88.

Zheng, Y.-T., Zhao, M., Song, Y., Adam, H., Buddemeier, U., Bissacco, A., Brucher, F., Chua, T.-S., and Neven, H. (2009). Tour the world: Building a web-scale landmark recognition engine. In *IEEE Conference on Computer Vision and Pattern Recognition, 2009. CVPR 2009*, pages 1085–1092.

Zhu, X., Ghahramani, Z., and Lafferty, J. (2003). Semi-supervised learning using Gaussian fields and harmonic functions. In *Proceedings of the 20th International Conference on Machine Learning (ICML-2003) Volume 2*, volume 2, pages 912–919, Washington, DC, US. AIAA Press.

Zigel, Y., Litvak, D., and Gannot, I. (2009). A method for automatic fall detection of elderly people using floor vibrations and sound #x2014;proof of concept on human mimicking doll falls. *IEEE Transactions on Biomedical Engineering*, 56(12):2858–2867.

Zinkevich, M., Weimer, M., Li, L., and Smola, A. J. (2010). Parallelized stochastic gradient descent. In *Advances in neural information processing systems*, pages 2595–2603.

Zivkovic, Z. (2004). Improved adaptive Gaussian mixture model for background subtraction. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004*, volume 2, pages 28–31 Vol.2.

Zivkovic, Z. and van der Heijden, F. (2006). Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern Recognition Letters*, 27(7):773–780.