

VIDEO PROCESSING ALGORITHMS FOR WILDFIRE SURVEILLANCE

A DISSERTATION SUBMITTED TO
THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE
OF BILKENT UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR
THE DEGREE OF
DOCTOR OF PHILOSOPHY
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

By
Osman Günay
May, 2015

Video Processing Algorithms for Wildfire Surveillance

By Osman Günay

May, 2015

We certify that we have read this dissertation and that in our opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.

Prof. Dr. A. Enis Çetin (Advisor)

Prof. Dr. Orhan Arıkan

Prof. Dr. Uğur Güdükbay

Assoc. Prof. Dr. Ali Ziya Alkar

Assist. Prof. Dr. B. Uğur Töreyn

Approved for the Graduate School of Engineering and Science:

Prof. Dr. Levent Onural
Director of the Graduate School

In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of Bilkent University's products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publicationsstandards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

Copyright Information

©2012 IEEE. Reprinted, with permission, from O. Gunay, B.U. Toreyin, K. Kose, and A. Enis Cetin, "Entropy-Functional-Based Online Adaptive Decision Fusion Framework With Application to Wildfire Detection in Video", IEEE Transactions on Image Processing, January 2012.

©2012 IEEE. Reprinted, with permission, from O. Gunay, B.U. Toreyin, K. Kose, and A. Enis Cetin, "Entropy functional based adaptive decision fusion framework", Signal Processing and Communications Applications Conference (SIU), 2012.

©2011 SPIE. Reprinted, with permission, from O. Gunay, B. Ugur Toreyin and A. Enis Cetin, "Online adaptive decision fusion framework based on projections onto convex sets with application to wildfire detection in video", Optical Engineering, July 2011.

©2011 EURASIP. Reprinted, with permission, from O. Gunay, H. Habiboglu and A. Enis Cetin, "Real-time wildfire detection using correlation descriptors", EUSIPCO, 2011.

©2014 Elsevier. Reprinted, with permission, from K. Kose, O. Gunay and A. Enis Cetin, "Compressive sensing using the modified entropy functional", Digital Signal Processing, January 2014.

©2013 Elsevier. Reprinted, with permission, from A. Enis Cetin et al., "Video fire detection - Review", Digital Signal Processing, December 2013.

ABSTRACT

VIDEO PROCESSING ALGORITHMS FOR WILDFIRE SURVEILLANCE

Osman Günay

Ph.D. in Electrical and Electronics Engineering

Advisor: Prof. Dr. A. Enis Çetin

May, 2015

We propose various image and video processing algorithms for wildfire surveillance. The proposed methods include; classifier fusion, online learning, real-time feature extraction, image registration and optimization. We develop an entropy functional based online classifier fusion framework. We use Bregman divergences as the distance measure of the projection operator onto the hyperplanes describing the output decisions of classifiers. We test the performance of the proposed system in a wildfire detection application with stationary cameras that scan predefined preset positions. In the second part of this thesis, we investigate different formulations and mixture applications for passive-aggressive online learning algorithms. We propose a classifier fusion method that can be used to increase the performance of multiple online learners or the same learners trained with different update parameters. We also introduce an aerial wildfire detection system to test the real-time performance of the analyzed algorithms. In the third part of the thesis we propose a real-time dynamic texture recognition method using random hyperplanes and deep neural networks. We divide dynamic texture videos into spatio-temporal blocks and extract features using local binary patterns (LBP). We reduce the computational cost of the exhaustive LBP method by using randomly sampled subset of pixels in the block. We use random hyperplanes and deep neural networks to reduce the dimensionality of the final feature vectors. We test the performance of the proposed method in a dynamic texture database. We also propose an application of the proposed method in real-time detection of flames in infrared videos. Using the same features we also propose a fast wildfire detection system using pan-tilt-zoom cameras and panoramic background subtraction. We use a hybrid method consisting of speeded-up robust features and mutual information to register consecutive images and form the panorama. The next step for multi-modal surveillance applications is the registration of images obtained with different devices. We propose a multi-modal image registration algorithm for infrared and visible range cameras. A new similarity measure is

described using log-polar transform and mutual information to recover rotation and scale parameters. Another similarity measure is introduced using mutual information and redundant wavelet transform to estimate translation parameters. The new cost function for translation parameters is minimized using a novel lifted projections onto convex sets method.

Keywords: Projections onto convex sets, classifier fusion, online learning, entropy maximization, wildfire detection, adaptive filtering, LMS, Bregman divergence, image Processing, infrared, mutual information, wavelet transform, image registration, log-polar transform, supporting hyperplanes..

ÖZET

ORMAN YANGINI GÖZETLEME AMAÇLI VIDEO İŞLEME ALGORİTMALARI

Osman Günay
Elektrik-Elektronik Mühendisliği, Doktora
Tez Danışmanı: Prof. Dr. A. Enis Çetin
Mayıs, 2015

Orman yangını gözetleme amaçlı çeşitli görüntü ve video işleme algoritmaları geliştirilmiştir. Önerilen yöntemler; sınıflandırıcı tümleştirme, çevrimiçi öğrenme, gerçek zamanlı öznitelik çıkarma, görüntü çakıştırma ve optimizasyonu içermektedir. İlk bölümde entropi fonksiyonu tabanlı çevrimiçi sınıflandırıcı tümleştirme altyapısı geliştirilmiştir. Bregman uyumsuzluğu uzaklık ölçüsü olarak kullanılarak, sınıflandırıcıların çıkış kararlarını açıklayan hiperdüzlemlerin üzerine izdüşümler yapılmıştır. Bu sistemin performansı, önceden tanımlı pozisyonları tarayan sabit kameralar ile orman yangını bulma yönteminde test edilmiştir. Tezin ikinci bölümünde, pasif-agresif çevrimiçi öğrenme algoritmalarının farklı formülasyonları ve karışım uygulamaları araştırılmıştır. Birden fazla çevrimiçi sınıflandırıcı veya farklı güncelleme parametreleri ile eğitilmiş aynı sınıflandırıcıların performansını artırmak için kullanılabilecek bir tümleştirme yöntemi önerilmiştir. Bu yöntemlerin gerçek zamanlı performansını test etmek için hava platformlarından video tabanlı orman yangını algılama sistemi tanıtılmıştır. Tezin üçüncü bölümünde, rastgele hiperdüzlemler ve derin sinir ağları kullanılarak gerçek zamanlı dinamik doku tanıma yöntemi önerilmiştir. Dinamik doku videoları uzam-zamansal bloklara bölünüp, yerel ikili örüntü (LBP) yöntemi kullanılarak öznitelikler çıkarılmıştır. Hesaplamalarda bloklardaki piksellerin rasgele seçilen bir altkümesi kullanılarak LBP metodunun hesap yükü azaltılmıştır. Rasgele hiperdüzlemler ve derin sinir ağları kullanılarak en son öznitelik vektörünün de boyutları azaltılmıştır. Bu yöntemin başarımı bir dinamik doku veri tabanında ve kızılötesi videolarda yangın tespiti probleminde test edilmiştir. Aynı öznitelikler ayrıca hareketli kameralar ve panoramik arkaplan kestirimi ile hızlı orman yangını tespiti için de kullanılmıştır. Gerçek zamanlı görüntülerin arkaplan görüntüsü ile çakıştırılması için hızlandırılmış gürbüz öznitelikler ve karşılıklı bilgi miktarı kullanılmıştır. Çok-kipli gözetleme uygulamaları için bir sonraki adım farklı cihazlar ile elde edilen görüntülerin

çakıştırılmasıdır. Bunun için kızılötesi ve görünür aralık kameralar için çok-kipli görüntü çakıştırma algoritması önerilmiştir. Dönme ve ölçek parametrelerini bulmak için log-polar dönüşümü ve karşılıklı bilgiye dayalı bir benzerlik ölçüsü tanımlanmıştır. Öteleme parametrelerini bulmak için ise karşılıklı bilgi ve artık dalgacık dönüşümü kullanılarak yeni bir benzerlik ölçüsü tanımlanmıştır. Bu benzerlik fonksiyonunu optimize etmek için dışbükey kümelerle izdüşüm yöntemine dayalı yeni bir yöntem geliştirilmiştir.

Anahtar sözcükler: İçbükey kümelerle izdüşüm, sınıflandırıcı birleşimi, çevrimiçi öğrenme, entropi maksimizasyonu, yangın algılama, adaptif filtreleme, LMS, Bregman ayrıklığı, görüntü işleme, kızılötesi, karşılıklı bilgi, dalgacık dönüşümü, görüntü eşleştirme, destek hiperdüzlemler, log-polar dönüşümü..

Acknowledgement

I would like to thank my supervisor Prof. A. Enis Çetin for his support and contribution in all stages of my academic life.

I am grateful to Prof. Dr. Orhan Arıkan and Prof. Dr. Uğur GÜDÜKBAY for accepting to be in my Ph.D. progress and defense committee and their support throughout my Ph.D. studies.

I would also like to thank Assist. Prof. Dr. B. Uğur TÖREYİN and Assoc. Prof. Dr. Ali Ziya Alkar for their valuable contributions and reviewing this thesis.

I would like to thank the administrative staff and personnel of General Directorate of Forestry, especially Mr. Nurettin Doğan, Mr. İlhami Aydın, for their collaboration.

This work was supported in part by the Scientific and Technological Research Council of Turkey (TÜBİTAK) with grant numbers 111E057, 105E191, 106G126.

This work was supported in part by the European Commission 7th Framework Program under Grant FP7-ENV-2009-1244088 FIRESENSE (Fire Detection and Management through a Multi-Sensor Network for the Protection of Cultural Heritage Areas from the Risk of Fire and Extreme Weather Conditions).

I would like to thank the Scientific and Technological Research Council of Turkey (TÜBİTAK) Science Fellowships and Grant Programmes Department (BİDEB) for my scholarship.

I wish to thank all of my friends and colleagues for their collaboration and support.

I am indebted to my wife and my family for their continuous support and encouragement throughout my life.

Contents

1	Introduction	1
1.1	Entropy Functional Based Online Classifier Fusion Framework . .	2
1.2	Generalized Update Equations for Online Adaptive Learning . . .	3
1.3	Real-time Dynamic Texture Recognition using Random Hyper-plane Projections	3
1.4	Wildfire Detection with PTZ Cameras using Panoramic Backgrounds	4
1.5	Multi-modal Image Registration using Mutual Information and Wavelet Measures	5
2	Entropy Functional Based Online Classifier Fusion Framework with Application to Wildfire Detection in Video	7
2.1	Introduction	7
2.2	Adaptive Classifier Fusion Algorithms	8
2.2.1	Weight Update Algorithm based on Orthogonal Projections	8
2.2.2	Entropic Projection (E-Projection) Based Weight Update Algorithm	9
2.2.3	Block Projection Method	13
2.3	Wildfire Detection Application	13
2.4	Weak Classifiers for Wildfire Detection Algorithm	15
2.5	Experimental Results	16
2.5.1	Experiments on wildfire detection	16
3	Generalized Update Equations for Online Adaptive Learning	22
3.1	Introduction	22
3.2	Online Learning Review	23

3.3	Online Learning Algorithms	24
3.3.1	Kernel Versions of Adaptive Algorithms	27
3.3.2	Mixture of Online Learners	28
3.4	Wildfire Detection with Online Classifiers	29
3.4.1	Image Segmentation	31
3.4.2	Color Analysis	34
3.4.3	Feature Extraction	34
3.5	Experimental Results	35
3.5.1	Experiments on Synthetic and Standard Datasets	35
3.5.2	Wildfire Detection Experiments	42
4	Real-time Dynamic Texture Recognition using Random Hyperplane Projections	45
4.1	Review	47
4.1.1	Random Hyperplanes and Deep Neural Networks	47
4.1.2	Local Binary Patterns	48
4.2	Feature Extraction and Classification	48
4.3	Infrared Flame Detection Algorithm	49
4.4	Experimental Results	50
4.4.1	Dynamic Texture Recognition Experiments	50
4.4.2	Infrared Flame Detection Experiments	58
5	Wildfire Detection with PTZ Cameras using Panoramic Backgrounds	64
5.1	Introduction	64
5.2	Panorama Generation	65
5.2.1	Robust Features	65
5.2.2	Mutual Information	66
5.2.3	Finding Key-frames	67
5.3	Wildfire Detection Algorithm	71
5.3.1	Feature Extraction	72
5.4	Experimental Results	73
6	Multi-modal Image Registration using Mutual Information and	

Wavelet Measures	77
6.1 Introduction	77
6.2 IR and Visible Image Registration	79
6.3 Cost Function and Optimization for Translations	81
6.3.1 Cost Function	81
6.3.2 Optimization Algorithm	87
6.4 Scale and Rotation Estimation	90
6.5 Experimental Results	93
7 Conclusion	100
Bibliography	101
Appendices	117
A Selection of Distance Function for Mutual Information	118
B Convex Optimization Algorithms based on Projections onto Convex Sets	121
B.1 Optimization Algorithm	123
B.1.1 Method 1: Projections onto Base Hyperplane	123
B.1.2 Method 2: Projections onto the Negative Function	125
B.2 Experimental Results	126
B.2.1 Polynomial Function Minimization Experiments	127
B.2.2 Logistic Regression Experiments	129
B.2.3 Image Registration Experiments	133

List of Figures

2.1	Flowchart of the weight update algorithm for one image frame. . .	15
2.2	Snapshots from the test videos in Table 2.1. The first two and the last two images are from the same video sequences.	17
2.3	False alarms issued to videos from Table 2.3. The first two and the last two images are from the same video sequences. Cloud shadows, clouds, fog, moving tree leaves, and sunlight reflecting from buildings cause false alarms	19
2.4	Average squared pixel errors for the NLMS and the ECF based algorithms for the video sequence $V12$	20
2.5	Adaptation of weights in a video that do not contain smoke. . . .	21
3.1	Fusion of Online Learners	29
3.2	Wildfire detection algorithm flowchart.	31
3.3	Example segmentation results for actual wildfire smoke images. . .	33
3.4	Average errors for different online learning algorithms for a synthetic dataset without noise.	36
3.5	Average errors for different online learning algorithms for a synthetic dataset with 8% label noise and instance noise ($\sigma_n = 0.1$). .	36
3.6	Average errors of OL-LMS algorithm for different values of the update parameter, μ and the result of the learner combination. . .	37
3.7	Average errors of OL-LMS algorithm for different values of the update parameter, μ and the result of the learner combination. . .	38
3.8	Average errors of kernel based algorithms on D01 dataset.	39
3.9	Average errors of kernel based algorithms on Vid10 sequence. . . .	41

3.10	Sample image frames from video sequences. Red rectangles mark the pedestrians, green rectangles mark the background samples. . .	41
3.11	Example detection results for actual wildfire smoke images.	44
4.1	LWIR and visible range images of long range flame; a)LWIR image, and b) visible range image.	47
4.2	Overview of the proposed feature extraction method.	49
4.3	<i>Alpha</i> dataset from DynTex database [1].	51
4.4	<i>Beta</i> dataset from DynTex database [1].	53
4.5	<i>Gamma</i> dataset from DynTex database [1].	55
4.6	Effect of number of random samples on success rate.	57
4.7	Effect of dimension of the random hyperplanes on success rate. . .	57
4.8	Effect of dimension of the last DNN layer on success rate.	58
4.9	Sample snapshots from flame detection dataset.	62
5.1	a) Key-frames, and b) resulting panorama from a sample video clip. Key-frames are marked with rectangles in the panorama. The video is taken from [2].	68
5.2	a) Key-frames, and b) resulting panorama in cylindrical coordinates from a sample video clip. Key-frames are marked with rectangles in the panorama. The video is taken from [2].	69
5.3	Wildfire surveillance images: a) current frame, and b) next frame.	70
5.4	a) Key-frames, and b) resulting panorama in cylindrical coordinates from a sample video. Key-frames are marked with rectangles in the panorama.	71
5.5	Frame index versus key-frame index for real-time operation using the background panorama of Figure 5.4: a) SURF method, b) MI method.	71
5.6	Overview of the proposed panoramic wildfire detection and feature extraction method.	73

5.7	Wildfire detection results for two different frame and key-frame pairs using the video sequence from Figure 5.4. Top-left image is the current frame, top-right image is the corresponding key-frame, bottom-left image is the registered current image, bottom-right image is the candidate smoke region.	76
5.8	Wildfire detection results for two different frame and key-frame pairs using the 360° coverage video sequence. Top-left image is the current frame, top-right image is the corresponding key-frame, bottom-left image is the registered current image, bottom-right image is the candidate smoke region.	76
6.1	Flowchart of the proposed method.	81
6.2	a) Original and translated visible range images, b) the joint histogram of unregistered images, and c) the joint histogram of registered images.	82
6.3	a) Original and translated visible range and LWIR images, b) the joint histogram of unregistered images, and c) the joint histogram of registered images.	83
6.4	Similarity value vs scale for a) DT-CWT, b) FSIM, and c) RWT similarity measures. The images that are used to calculate the measures are given in Figure 6.3.	84
6.5	Similarity value vs xy-translations for a) DT-CWT, b) FSIM, and c) RWT similarity measures. Images that are used to calculate the measures are given in Figure 6.3.	85
6.6	1-Level redundant wavelet transform.	86
6.7	a) Minimization of a convex function by projections when closed form expression of the projection operator is available. b) Minimization of a convex function by projections onto supporting hyperplanes of the cost function.	88
6.8	a) Reference image (from [3]), b) log-polar transforms of reference image, c) scaled/rotated image, d) log-polar transform of scaled/rotated image.	91

6.9	a) Visible range image b) IR image, c) FFT of visible range image, d) FFT of IR images, e) log-polar transform of FFT of visible image, f) log-polar transform of FFT of IR image.	92
6.10	Mutual information cost value vs scale and rotations.	93
6.11	a) Visible range image b) IR image, c) joint histogram for unregistered images, d) joint histogram for registered images, e) cost function vs translations.	95
6.12	a) Evolution of transformation parameters vs iterations, b) evolution of the cost function vs iterations.	96
6.13	Registration results for Image Set 1: a) visible Image, b) IR image, c) scaled and rotate IR image, translated IR image.	99
A.1	Distance functions.	119
B.1	a) Minimization of a convex function by projections when closed form expression of the projection operator is available, b) minimization of a convex function by projections onto supporting hyperplanes of the cost function.	124
B.2	Correction method for projections onto supporting hyperplanes. .	125
B.3	a) Minimization of a convex function by projections when closed form expression of the projection operator is available, b) minimization of a convex function by projections onto supporting hyperplanes of the cost function.	126
B.4	Function value versus iterations for $f_3(\mathbf{x})$, a) iterations 0-50, b) iterations, 950-1000, c) iterations 990-1000.	129
B.5	Function value versus iterations for diabetes_scale dataset, a) iterations 0-50, b) iterations, 50-100, c) iterations 150-200.	133
B.6	Function value versus iterations for third image set, a) iterations 0-16, b) iterations, 17-32, c) iterations 33-50.	137
B.7	Example image registration result for Iowa Landsat images [4], a) band 1, b) band 5, c) registered band 5 image.	138
B.8	Example image registration result for TerraSAR-X and Ikonos images [5], a) Ikonos image, b) TerraSAR-X image, c) registered TerraSAR-X image.	138

B.9	Example image registration result for MTI images [6], a) band A image, b) band D image, c) registered band D image.	139
B.10	Mutual information cost function versus transformation parameters, a) TerraSAR-X/Ikonos images [5], b) MTI images [6].	140
B.11	Mutual information cost function versus iteration for different initial shift parameters on the fourth Landsat image set: a) quasi-Newton method, b) POCS1 method.	141
B.12	Mutual information cost function versus iteration for different initial shift parameters on the third Landsat image set: a) quasi-Newton method, b) POCS1 method.	142

List of Tables

2.1	Eight different algorithms are compared in terms of true detection rates.	18
2.2	Eight different algorithms are compared in terms of first alarm frames and times.	18
2.3	Eight different algorithms are compared in terms of false alarm rates.	19
3.1	Descriptions of datasets used in kernel experiments.	39
3.2	Success rates of kernel based algorithms on UCI machine learning datasets.	40
3.3	Success rates of kernel based algorithms on thermal pedestrian dataset.	42
3.4	Test results on wildfire detection experiment.	43
3.5	Average processing times (in μs) of online learning algorithms on the first video sequence.	44
4.1	Dynamic texture recognition success rates (percentage) on <i>Alpha</i> dataset. In the ratio NS/A, NS is the number of total samples and A is the ratio of samples used for training. CV column displays the 10-fold cross-validation accuracy. c_i is the number of samples for the i_{th} class.	52
4.2	Dynamic texture recognition success rates (percentage) on <i>Beta</i> dataset. In the ratio NS/A, NS is the number of total samples and A is the ratio of samples used for training. CV column displays the 10-fold cross-validation accuracy. c_i is the number of samples for the i_{th} class.	54

4.3	Dynamic texture recognition success rates (percentage) on <i>Gamma</i> dataset. In the ratio NS/A, NS is number of total samples and A is the ratio of samples used for training. CV column displays the 10-fold cross-validation accuracy. c_i is the number of samples for the i_{th} class.	56
4.4	Dynamic texture recognition success rates (percentage) on infrared flame dataset. In the ratio NS/A, NS is the number of total training samples and A is the ratio of the training set that is used. CV column displays the 10-fold cross-validation accuracy on the whole dataset. c_i is the number of samples for the i_{th} class.	59
4.5	<i>Number of support vectors total classification time (sec)</i> for flame detection dataset.	59
4.6	Proposed method is compared with [7] in terms of correct detection rates.	60
4.7	Proposed method is compared with [7] in terms of false detection rates	61
5.1	Wildfire smoke recognition success rates (percentage). In the ratio NS/A, NS is the total number of training samples and A is the ratio of the training set that is used. c_i is the number of samples for the i_{th} class.	74
5.2	<i>Number of support vectors total classification time (sec)</i> for smoke detection dataset.	75
5.3	LBP based method compared to methods in Chapters 2 and 3 on a common dataset.	75
6.1	Registration results for the first experiment comparing gradient descent proposed methods. Here t_x and t_y represent x-y translations.	95
6.2	Registration results for the second experiment. Error is defined as: $[\Delta s \times 10 + \Delta \phi \times 10 + \Delta t_x + \Delta t_y]$. Δ represents the difference between the true value of the parameter and the optimization result.	96

A.1	Optimization errors (sum of the absolute differences between the actual parameter values and optimization results) for different cost functions on various image sets.	119
A.2	Optimization times (sec) for different cost functions on various image sets.	120
B.1	Optimization results for polynomial functions. Times are in seconds.	128
B.2	Training optimization and classification results for logistic regression experiments. Times are in seconds.	130
B.3	Multi-modal Image registration results. Times are in seconds. . .	135

Chapter 1

Introduction

We propose image and video processing methods for various wildfire surveillance applications. The main issue we deal with is the early detection of wildfire smoke with different imaging devices (fixed/moving visible range and infrared cameras). The most important problem for video based wildfire detection is the abundance of false alarms caused by changing environment conditions [8–10]. This necessitates the implementation of online learning algorithms that can adapt to sudden and continuous changes in the environment. We present two new online learning algorithms to solve this problem. The first algorithm is an online classifier fusion method based on entropic projections onto hyperplanes that describe the outputs of several weak classifiers that are trained offline. In this method, weak outputs of different classifier are linearly combined to achieve increased performance. This method is designed and tested with fixed pan-tilt-zoom (PTZ) cameras that scan programmed preset positions. The second algorithm is an online learning method for wildfire detection from moving aerial platforms. In this method, instead of combining the outputs of offline-trained classifiers we perform the training of each classifier online. We also propose a similar classifier combination method for online classifiers. For this problem we propose image segmentation and feature extraction methods for wildfire detection from moving cameras without using motion information.

Flame and smoke in video can be categorized as dynamic textures that are moving regions in image sequences but display some sort of temporal stationarity [11]. There are many methods in the literature for recognition of dynamic textures but

they are computationally expensive [12]. We propose real-time dynamic texture recognition methods using local binary patterns and random hyperplane projections. We present an application of the proposed method for fire detection in infrared videos. We also use this algorithm for wildfire detection using panoramic background with continuously moving cameras.

When different imaging devices are used for surveillance applications, it becomes necessary to register multi-modal images of imaging devices and present combined detection results. For this problem we propose an image registration method using mutual information and wavelet measures for matching coordinates of natural images. In the registration process we use a projection onto convex sets (POCS) based optimization method that does not require an update parameter and works by performing projections onto supporting hyperplanes of the cost function to be minimized. This method shows excellent performance in multi-modal image registration problem.

1.1 Entropy Functional Based Online Classifier Fusion Framework

We propose an online learning framework, called entropy functional based classifier fusion (ECF), which can be used in various video processing applications. We assume that the final decision is obtained from the linear combination of outputs of several weak classifiers. The weights of the algorithm are updated using entropic projections (e-projections) onto convex sets that represent weak classifiers.

Adaptive learning methods based on orthogonal projections are successfully used in some computer vision and pattern recognition problems [13, 14]. Instead of determining the weights using orthogonal projections as in [13, 14], we introduce the entropic e-projection approach which is based on a generalized projection onto a convex set.

The main contributions of this chapter are described below:

- We propose Bregman divergence based projection method for classifier fusion optimization; and we develop an online wildfire detection method for

forest surveillance applications.

1.2 Generalized Update Equations for Online Adaptive Learning

We investigate four maximum-margin online learning algorithms based on the passive-aggressive update strategy [15] using a generalized update function and a mixture-of-experts framework. We obtain the first three methods using the update strategies of stochastic gradient, recursive least squares and exponentiated gradient algorithms [16, 17]. The final algorithm is a mixture of experts (MOE) method which we obtain by linearly combining the outputs of different online learning algorithms or the same algorithm with different update parameters [18]. We use Bregman divergence as the distance function to obtain the weights of the mixture. Bregman divergences provide easier analytical analysis to obtain non-negative weights that sum up to unity. We also develop a real-time detection method that processes and classifies visible range image sequences recorded on aerial platforms for early wildfire warning systems.

The main contributions of this chapter are described below:

- We obtain passive-aggressive online learning algorithms from a generalized optimization problem and evaluate their performance on various binary classification problems.
- We use Bregman divergences to linearly combine the outputs of online learners to achieve better accuracy.
- We design a wildfire detection method for visible range cameras mounted on aerial platforms.

1.3 Real-time Dynamic Texture Recognition using Random Hyperplane Projections

We introduce new methods for applications of local binary patterns (LBP) in real-time dynamic texture recognition. LBPs are first introduced for recognition

of textures in images [19]. Later they are extended to temporal domain for classification of image sequences [20]. In the conventional methods, videos are divided into spatio-temporal blocks and LBP computation is performed for each pixel in the block. LBP consists of comparing center pixel to its neighbors and forming a binary number (1:larger than, 0:smaller than neighbor). The histograms of binary numbers form the feature vectors. When the number of neighbors increases, usually the descriptive ability of the feature also rises, but this also requires more computational power. We propose two improvements to speed-up feature extraction and classification process; i) instead of using every pixel in the block we use only a small subset of pixels at random locations, ii) we use random hyperplanes and deep neural networks to decrease the dimension of feature vectors. We test the performance of the proposed methods in DynTex dynamic texture database [1] and also in real-time infrared flame detection applications. The main contributions of this chapter are described below:

- We use random sampling to significantly reduce the computational cost of LBP-TOP (three orthogonal planes) method.
- We show that using random hyperplanes or deep neural networks to reduce the dimension of feature vectors can decrease computational cost and preserve classification accuracy.
- We propose a real-time infrared flame detection system using the proposed feature extraction methods.

1.4 Wildfire Detection with PTZ Cameras using Panoramic Backgrounds

We propose a real-time panoramic background estimation method for fast wildfire detection using continuously moving cameras. Panoramic background generation has many uses for image processing and surveillance applications. Background generation mainly depends on efficient registration of successive images in the panoramic sequence. Registration algorithms can either be feature based [21] or global [22]. Feature based methods try to match the locations of features

between the images, whereas global methods optimize a similarity measure to find a transformation matrix that best matches the images. The main contributions of this chapter are described below:

- We propose a hybrid method for registration of the images to the panoramic background using robust features and mutual information.
- We exploit robust features to characterize smoke behaviour.
- We adapt the LBP based dynamic texture recognition to wildfire smoke sequences.

1.5 Multi-modal Image Registration using Mutual Information and Wavelet Measures

With the decrease of the costs of infrared (IR) sensors, it is now possible to use IR cameras together with regular cameras in many practical systems. It is important to match IR and visible range camera images to obtain multi-modal detection results in many surveillance applications. IR and visible images have different intensity ranges, therefore it is not possible to use standard methods to register these images [23]. Early studies in multi-modal image registration have focused on registration of medical images obtained using different imaging devices (PET, CT, MRI, etc.) [22].

We propose different cost functions for estimation of scale/rotation and translation parameters respectively. We use mutual information of log-polar transformed Fourier transforms of multi-modal images to obtain scale and rotation parameters. We use mutual information and wavelet transform based cost function to obtain translation parameters.

The main contributions of this chapter are described below:

- We propose a mutual information based method for recovering scale and rotation difference between infrared and visible images.
- We propose a redundant wavelet transform based complementary cost function for recovering translation parameters. This cost function complements the lack of spatial information in mutual information.

- We propose a supporting hyperplane projection based optimization to minimize smooth cost functions without requiring an update parameter as in gradient descent approaches.

Chapter 2

Entropy Functional Based Online Classifier Fusion Framework with Application to Wildfire Detection in Video

2.1 Introduction

Multiple weak classifier system can outperform one strong classifier especially in applications with changing environments [24]. We propose a classifier fusion system based on Bregman divergence and Shannon entropy. The proposed entropy functional based classifier fusion (ECF) framework is used to improve the orthogonal projection based wildfire detection methods proposed in [10, 25]. The algorithm is based on five weak classifiers: (i) moving object detection, (ii) color analysis, (iii) wavelet analysis, (iv) shadow analysis, (v) covariance matrix based classification. Outputs of classifiers represent the confidence of their algorithm in determining the existence of wildfire and they are combined using the ECF method. The weights of the classifier outputs are updated using entropic e-projections onto hyperplanes which denote the convex sets corresponding to each classifier. This method can be categorized as a supervised online learning problem. The supervisor for the wildfire detection problem is the security guard at the lookout tower.

The results in this chapter was previously published in [9]. The rest of the chapter is organized as follows: ECF framework is described in Section 2.2. The first part of this section describes our previous weight update algorithm which is obtained by orthogonal projections onto hyperplanes [13, 25, 26], the second part proposes an entropy based e-projection method for weight update. Section 2.3 introduces the video based wildfire detection problem. In Section 2.4, five weak classifier for wildfire detection is reviewed. In Section 2.5, experimental results are presented. The proposed framework is not restricted to the wildfire detection problem.

2.2 Adaptive Classifier Fusion Algorithms

Assume we have M classifiers with outputs denoted as, $\mathbf{x}_t = [\mathbf{x}_t(1), \dots, \mathbf{x}_t(M)]^T$, time step t . We denote the weights of classifiers as $\mathbf{w}_t = [\mathbf{w}_t(1), \dots, \mathbf{w}_t(M)]^T$. The output of the linear classifier combination can be expressed at time t as

$$\hat{y}_t = \mathbf{x}_t^T \mathbf{w} = \sum_i \mathbf{w}_t(i) \mathbf{x}_t(i). \quad (2.1)$$

This is an estimate of the true binary class label y_t . The error is also defined as $e_t = y_t - \hat{y}_t$.

2.2.1 Weight Update Algorithm based on Orthogonal Projections

In this section, we review the orthogonal projection based weight update scheme [10, 13]. The correct classification result y_t can be represented as an M -dimensional hyperplane:

$$y_t = \mathbf{x}_t^T \mathbf{w}_t. \quad (2.2)$$

In these methods, the weights are updated by finding the projection of the current weight vector $\mathbf{w}_t$ onto the hyperplane in Equation 2.2.

Orthogonal projections can be represented as the following optimization problem

$$\begin{aligned} \min_{\mathbf{w}^*} & \|\mathbf{w}^* - \mathbf{w}_t\|_2, \\ \text{subject to } & \mathbf{x}_t^T \mathbf{w}^* = y_t. \end{aligned} \quad (2.3)$$

The solution is also called the metric projection mapping solution. However we use the term orthogonal projection because the line going through $\mathbf{w}^*$ and $\mathbf{w}_t$ is

orthogonal to the hyperplane. The updated weights $\mathbf{w}_{t+1} = \mathbf{w}^*$ can be obtained by the following iteration:

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \frac{e_t}{\|\mathbf{x}_t\|_2^2} \mathbf{x}_t. \quad (2.4)$$

We note that Equation 2.4 is similar to the normalized least mean square (NLMS) algorithm with update parameter $\mu = 1$. According to the projection onto convex sets (POCS) theory, when there are a finite number of convex sets, repeated cyclical projections onto these sets converge to a vector in the intersection set [27–31]. The case of an infinite number of convex sets is studied in [14, 32, 33]. They propose to use the convex combination of the projections onto the most recent q sets for online adaptive algorithms [14]. In Section 2.2.3, the block projection version of the algorithm that deals with the case when there are an infinite number of convex sets is presented.

In real-time operation when a new input is received at time step $t+1$, the following hyperplane can be defined $\mathbb{R}^M$

$$y_{t+1} = \mathbf{x}_{t+1}^T \mathbf{w}^*. \quad (2.5)$$

When there are a finite number of hyperplanes, iterated weights that are obtained by cyclic projections onto these hyperplanes converge to their intersection [27, 34, 35].

2.2.2 Entropic Projection (E-Projection) Based Weight Update Algorithm

We present the entropic projection based weight update algorithm. The findings in this chapter was previously published in [36]. We use Bregman divergence based projection methods in the weight update algorithm.

The e-projection is a generalized metric projection mapping onto a convex set [36, 37]. Let $\mathbf{w}_t$ denote the weight vector for the t_{th} sample. Its e-projection $\mathbf{w}^*$ onto a convex set C using the cost function $g(\mathbf{w})$ is:

$$\mathbf{w}^* = \arg \min_{\mathbf{w} \in C} L(\mathbf{w}, \mathbf{w}_t), \quad (2.6)$$

where

$$L(\mathbf{w}, \mathbf{w}_t) = g(\mathbf{w}) - g(\mathbf{w}_t) - \langle \nabla g(\mathbf{w}), \mathbf{w} - \mathbf{w}_t \rangle, \quad (2.7)$$

and $\langle ., . \rangle$ represents the inner product.

In the adaptive learning problem, we have a hyperplane $H_t : \mathbf{x}_t^T \mathbf{w}_{t+1} = y$ For each hyperplane H_t , the e-projection (Equation 2.6) is equivalent to

$$\nabla g(\mathbf{w}_{t+1}) = \nabla g(\mathbf{w}_t) + \lambda \mathbf{x}_t, \quad (2.8)$$

$$\mathbf{x}_t^T \mathbf{w}_{t+1} = y_t, \quad (2.9)$$

where the Lagrange multiplier λ should be determined. When the cost function is Euclidean $g(\mathbf{w}) = \sum_i \mathbf{w}_t(i)^2$ the distance $L(\mathbf{w}, \mathbf{v})$ becomes the l_2 norm square of the difference vector $(\mathbf{w} - \mathbf{v})$, and the e-projection becomes the orthogonal projection.

When we use entropy function $g(\mathbf{w}) = \sum_i \mathbf{w}_t(i) \log(\mathbf{w}_t(i))$ as the cost function, the e-projection onto the hyperplane H_t leads to the following update equations:

$$w_{t+1}(i) = w_t(i) e^{\lambda \mathbf{x}_t(i)}, \quad i = 1, 2, \dots, M. \quad (2.10)$$

The Lagrange parameter λ is determined using Equation 2.10 in the equation:

$$\mathbf{x}_t^T \mathbf{w}_{t+1} = y_t, \quad (2.11)$$

because the e-projection $\mathbf{w}_{t+1}$ must be on the hyperplane H_t in Equation 2.9.

To find the value of λ at each iteration a nonlinear equation has to be solved (Equations 2.10 and 2.11). In [38], globally convergent algorithms are developed without finding the exact value of the Lagrange multiplier λ . However, the tracking performance of the algorithm is very important. Weights have to be rapidly updated according to the user's decision.

In our application, we first use the second order Taylor series approximation of $e^{\lambda \mathbf{x}_t(i)}$ from Equation 2.10 and obtain:

$$\mathbf{w}_{t+1}(i) \approx \mathbf{w}_t(i) \left(1 + \hat{\lambda} \mathbf{x}_t(i) + \frac{\hat{\lambda}^2 \mathbf{x}_t(i)^2}{2} \right), \quad i = 1, 2, \dots, M. \quad (2.12)$$

Multiplying both sides by $\mathbf{x}_t(i)$, summing over i and using Equation 2.11 we get the following equation:

$$y_t \approx \sum_{i=1}^M \left(\mathbf{x}_t(i) \mathbf{w}_t(i) + \hat{\lambda} \sum_{i=1}^M \mathbf{x}_t(i)^2 \mathbf{w}_t(i) + \hat{\lambda}^2 \sum_{i=1}^M \frac{\mathbf{x}_t(i)^3 \mathbf{w}_t(i)}{2} \right).$$

We can solve for the initial value of λ from Equation 2.13 analytically. We insert the two solutions of Equation 2.13 into Equation 2.10 and pick the $\mathbf{w}_{t+1}$ vector closest to the hyperplane in Equation 2.11. This is determined by checking the error e_t . We experimentally observed that this estimate provides convergence in forest fire application. To determine a more accurate value of Lagrange multiplier λ we developed a heuristic search method based on the estimate $\hat{\lambda}$. If $e_t < 0$, we choose $\lambda_{min} = \hat{\lambda} - 2|\hat{\lambda}|$, $\lambda_{max} = \hat{\lambda}$ and if $e_t > 0$, we choose $\lambda_{min} = \hat{\lambda}$, $\lambda_{max} = \hat{\lambda} + 2|\hat{\lambda}|$ as the upper and lower bounds of the search window. We only look at R values uniformly distributed between these limits to find the best $\hat{\lambda}$ that produces the lowest error. We could have used a fourth order Taylor series approximation in Equation 2.12 and still obtained an analytical solution. After fourth order approximations, a solution has to be numerically found. There are very efficient polynomial root finding algorithms in the literature.

The algorithm for the e-projection based classifier fusion method is given in Algorithm 1, which explains projection onto one hyperplane. In the Algorithm λ_{min} and λ_{max} are determined from the Taylor series approximation as described above. The temporary variables $\mathbf{v}$ and $\mathbf{w}_T$ are used to find the λ value that produces the lowest error. A different λ value is determined for each sample at each time step. Obviously a new value of λ has to be computed whenever a new observation arrives.

Instead of the Shannon entropy $x \log x$, it is possible to use the regular entropy function $\log x$ as the cost functional [38]. In this case,

$$g(\mathbf{w}) = - \sum_i \log(\mathbf{w}_t(i)), \quad (2.13)$$

which is convex for $\mathbf{w}_t(i) > 0$. The e-projection onto the hyperplane H_t can be obtained as follows:

$$\mathbf{w}_{t+1}(i) = \frac{\mathbf{w}_t(i)}{1 + \lambda \mathbf{w}_t(i) \mathbf{x}_t(i)}, \quad i = 1, 2, \dots, M, \quad (2.14)$$

where the update parameter λ can again be obtained by inserting Equation 2.14 into the hyperplane constraint in Equation 2.11. Penalizing the $\mathbf{w}_t(i) = 0$ case with an infinite cost may not be suitable for online classifier fusion problems. However, the cost function:

$$g(\mathbf{w}) = - \sum_i \log(\mathbf{w}_t(i) + 1), \quad (2.15)$$

Algorithm 1 The algorithm ECF method

```

for  $i = 1$  to  $M$  do
     $\mathbf{w}_0(i) = \frac{1}{M}$ , Initialization
end for
For each sample at time step  $t$ .
for  $\lambda = \lambda_{min}$  to  $\lambda_{max}$  do
    for  $i = 1$  to  $M$  do
         $\mathbf{v}_t(i) = \mathbf{w}_t(i)$ 
         $\mathbf{v}_t(i) \leftarrow \mathbf{v}_t(i)e^{\lambda \mathbf{x}_t(i)}$ 
    end for
    if  $\|y_t - \sum_i \mathbf{v}_t(i)\mathbf{x}_t(i)\|_2 < \|y_t - \sum_i \mathbf{w}_t(i)\mathbf{x}_t(i)\|_2$  then
         $\mathbf{z}_t \leftarrow \mathbf{v}_t$ 
    end if
end for
 $\mathbf{w}_t \leftarrow \mathbf{z}_t$ 
for  $i = 1$  to  $M$  do
     $\mathbf{w}_t(i) \leftarrow \frac{\mathbf{w}_t(i)}{\sum_j \mathbf{w}_t(j)}$ 
end for
 $\hat{y}_t = \sum_i \mathbf{w}_t(i)\mathbf{x}_t(i)$ 
if  $\hat{y}_t \geq 0$  then
    return 1
else
    return -1
end if

```

is always positive, convex and differentiable for $\mathbf{w}_t(i) \geq 0$. In this case, weight update equation becomes:

$$\mathbf{w}_{t+1}(i) = \frac{\mathbf{w}_t(i) - \lambda(\mathbf{w}_t(i) + 1)\mathbf{x}_t(i)}{1 + \lambda(\mathbf{w}_t(i) + 1)\mathbf{x}_t(i)}, \quad i = 1, 2, \dots, M, \quad (2.16)$$

where the update parameter λ should be determined using by substituting Equation 2.16 into Equation 2.11. Finding the exact value of λ when Equation 2.11 is only a four dimensional hyperplane, using numerical methods is not difficult. In the forest fire detection problem we have only five weak classifiers. However, when the number of weak classifiers are high, new numerical methods should be determined for cost functions in Equations 2.13 and 2.15.

2.2.3 Block Projection Method

Block projection based methods are developed for inverse problems and active fusion methods [14, 32, 33, 39]. In this case, sets are assumed to arrive sequentially and q of the most recently received observation sets are used to update the weights in the block projection approach. Adaptive projected subgradient method (APSM) works by taking a convex combination of the projections of the current weight vector onto those q sets. The weights calculated using this method are shown to converge to the intersection of hyperplanes [14]; i.e., for each sample there exist $\mathbf{w}^*$ such that:

$$\mathbf{w}^* \in \bigcap_{t \geq t_0} H_t, \quad (2.17)$$

where $t_0 \in \mathbb{N}$.

The next values of weights $\mathbf{w}_{t+1}$ can be calculated from the q projections $P_{H_t(j)}(\mathbf{w}_t)$ for $j \in S_n = \{n - q + 1, n - q + 2, \dots, n\}$ using the APSM as follows:

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \mu_t \left(\sum_{j \in S_t} \alpha_t(j) P_{H_t(j)}(\mathbf{w}_t) - \mathbf{w}_t \right), \quad (2.18)$$

where $\alpha_t(j)$ is a weight used to control the contribution of the projection onto j th hyperplane and $\sum_{j \in S_t} \alpha_t(j) = 1$, any μ_t can be chosen from $(0, 2\mathcal{M}_t)$ where:

$$\mathcal{M}_t = \frac{\sum_{j \in S_t} \alpha_t(j) \|P_{H_t(j)}(\mathbf{w}_t) - \mathbf{w}_t\|_2}{\left\| \sum_{j \in S_t} \alpha_t(j) P_{H_t(j)}(\mathbf{w}_t) - \mathbf{w}_t \right\|_2}. \quad (2.19)$$

The weights of projections are usually chosen as $\alpha_t(j) = 1/q$ and μ_t can be chosen as 1 since $\mathcal{M}_t \geq 1$ is always true [14]. Both orthogonal and entropic projections can be used as the projection operator, $P_{H_t(j)}$. We experimentally observed the convergence of the entropic method.

2.3 Wildfire Detection Application

We present an application of the proposed ECF method for video based wildfire detection. In the long range, wildfire detection scenarios smoke becomes visible before the flames and it is important to detect the existence of smoke for early wildfire detection [40, 41]. From our extensive experiments we observed that

the best way to detect smoke is using visible range cameras. LWIR and SWIR cameras see through smoke and they can only perform detection when the flames of wildfire become visible.

Most surveillance systems already have built-in simple detection modules (e.g. motion detection, event analysis). Recently, there is also significant interest in developing real-time algorithms to detect fire and smoke for standard surveillance systems [26,42]. Smoke is difficult to model due to its dynamic texture and irregular motion characteristics. Unstable cameras, dynamic backgrounds, obstacles in the range of the camera and lighting conditions also pose important problems for smoke detection. Smoke plume observed from a long distance and observed from up close have different spatial and temporal characteristics. Therefore, generally different algorithms are designed to detect close range and long range smoke plume. Jerome and Philippe [42, 43] implemented a real-time automatic smoke detection system for forest surveillance stations. The main assumption for their detection method is that the energy of the velocity distribution of smoke plume is higher than other natural occurrences except for clouds which, on the other hand have lower standard deviation than smoke. In the classification stage they use fractal embedding and linked list chaining to segment smoke regions. This method was used in the forest fire detector “ARTIS FIRE”, commercialized by “T2M Automation”. Another smoke detection method with an application to wildfire prevention was described in [44]. This method takes the advantages of wavelet decomposition and optical flow algorithm for fire smoke detection and monitoring. The optical flow algorithm is used for motion detection. Wavelet decomposition based method was used to solve the aperture problem in optical flow. After the smoke is detected and segmented, smoke characteristics such as speed, dispersion, apparent volume, maximum height, gray level and inclination angle of the smoke can be extracted using the video frames or image sequences. Qinjuan et al. [45] proposed a method for long range smoke detection to be used in a wildfire surveillance system. The method uses multi-frame temporal difference and OTSU thresholding to find the moving smoke regions. They also use color and area growth clues to verify the existence of smoke.

2.4 Weak Classifiers for Wildfire Detection Algorithm

The proposed wildfire detection method contains of five weak classifiers: (i) moving object detection, (ii) color analysis, (iii) wavelet analysis, (iv) shadow analysis, (v) covariance matrix based classification, with output values, $x_t(1)$, $x_t(2)$, $x_t(3)$, $x_t(4)$ and $x_t(5)$, respectively.

The first four algorithms are described in detail in [10] which is available online at the EURASIP webpage. We recently added the fifth algorithm to our system whose details are given in [9, 25].

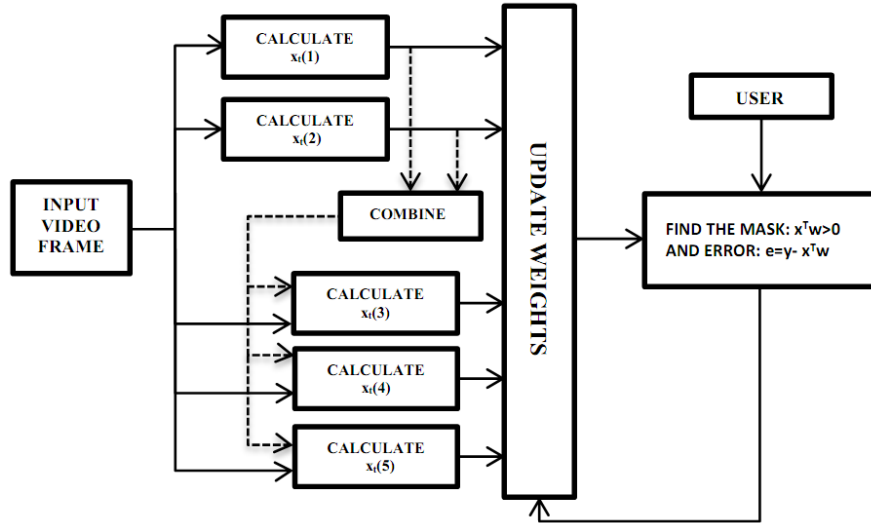


Figure 2.1: Flowchart of the weight update algorithm for one image frame.

The flowchart of the wildfire detection system is given in Figure 2.1. In the wildfire detection system installed in watch towers, PTZ cameras continuously move between preset positions. We keep separate weights for each preset and update their weights independently. In this way the weights are specialized to each field of view of camera and we can reduce false alarms.

2.5 Experimental Results

2.5.1 Experiments on wildfire detection

In the experiments the proposed ECF method and the universal linear predictor (ULP) method [46] are compared. ULP method updates the weights of i_{th} classifier ($\mathbf{v}_t(i)$) as follows:

$$\mathbf{v}_t(i) = \frac{\exp(-\frac{1}{2\alpha}(y_t - \mathbf{x}_t(i))^2)}{\sum_j \exp(-\frac{1}{2\alpha}(y_t - \mathbf{x}_t(j))^2)}, \quad (2.20)$$

where α is a constant update parameter.

In the experiments, we compared eight different algorithms named FIXED, ULP, NLMS, NLMS-B, ECF, ECF-B, LOGX and LOG(X+1). NLMS-B and ECF-B are block projection versions of NLMS and ECF based methods with block size $q = 5$. LOGX and LOG(X+1) represent the algorithms that use $-\log x$ and $-\log(x+1)$ as the distance functions. FIXED represents the unadaptive method that uses fixed weights and ULP is the universal linear predictor based approach.

results in Table 2.1 are given in Figure 2.2.

Table 2.1: Eight different algorithms are compared in terms of true detection rates.

Video	Frames	True Detection Rates							
		FIXED	ULP	NLMS	NLMS-B	ECF	ECF-B	LOGX	LOG(X+1)
V1	768	87.63%	87.63%	87.63%	87.63%	87.63%	87.63%	87.89%	87.63%
V2	300	89.67%	89.67%	83.00%	89.66%	81.33%	86.00%	84.67%	89.66%
V3	550	70.36%	70.36%	68.18%	68.18%	67.09%	68.18%	67.09%	68.00%
V4	1000	94.90%	94.90%	90.80%	94.10%	90.50%	92.40%	93.30%	93.70%
V5	1000	96.30%	95.50%	91.10%	92.90%	91.90%	92.70%	92.40%	93.40%
V6	439	80.87%	80.87%	80.41%	80.41%	80.41%	80.41%	80.41%	80.41%
V7	770	85.71%	85.71%	85.71%	85.71%	85.84%	85.71%	85.71%	85.97%
V8	1060	98.68%	99.15%	98.86%	98.68%	98.77%	98.67%	98.96%	98.77%
V9	410	80.24%	80.24%	80.00%	80.00%	80.00%	80.00%	80.00%	80.00%
V10	1000	82.30%	82.30%	79.30%	82.40%	89.50%	90.70%	91.10%	81.30%
Avg.	-	86.67%	86.63%	84.50%	85.97%	85.30%	86.24%	86.15%	85.88

Table 2.2: Eight different algorithms are compared in terms of first alarm frames and times.

Video	First Alarm Frame / Time (secs.)							
	FIXED	ULP	NLMS	NLMS-B	ECF	ECF-B	LOGX	LOG(X+1)
V1	64/12.80	64/12.80	64/12.80	64/12.80	64/12.80	64/12.80	64/12.80	64/12.80
V2	42/8.40	42/8.40	67/13.40	42/8.40	68/13.60	53/10.60	58/11.60	42/8.40
V3	26/5.20	26/5.20	37/7.40	37/7.40	44/8.80	37/7.40	43/8.60	38/7.60
V4	25/5.00	25/5.00	58/11.60	25/5.00	59/11.80	33/6.60	25/5.00	43/8.60
V5	32/6.40	35/7.00	53/10.60	35/7.00	54/10.80	35/7.00	35/7.00	36/7.20
V6	21/4.20	21/4.20	21/4.20	21/4.20	21/4.20	21/4.20	21/4.20	21/4.20
V7	47/1.88	47/1.88	47/1.88	47/1.88	47/1.88	47/1.88	47/1.88	47/1.88
V8	12/1.33	12/1.33	12/1.33	12/1.33	12/1.33	12/1.33	12/1.33	12/1.33
V9	67/2.68	67/2.68	67/2.68	67/2.68	67/2.68	67/2.68	67/2.68	67/2.68
V10	33/6.60	33/6.60	50/10.00	33/6.60	51/10.20	33/6.60	33/6.60	44/8.80
Avg.	36.90/5.45	37.20/5.51	47.60/7.59	38.30/5.73	48.70/7.81	40.20/6.11	40.50/6.17	41.40/6.35

In Table 2.3 the algorithms are compared in terms of false alarm rates. Except for one video sequence, ECF method produces the lowest false alarm rate in the dataset. The algorithms that use adaptive fusion strategy significantly reduce the false alarm ratio of the system. One interesting result is that ECF-B and NLMS-B, which are the versions that use the block projection method developed for the case of infinite number of convex sets, usually produced more false alarms than the methods that do not use block projections.

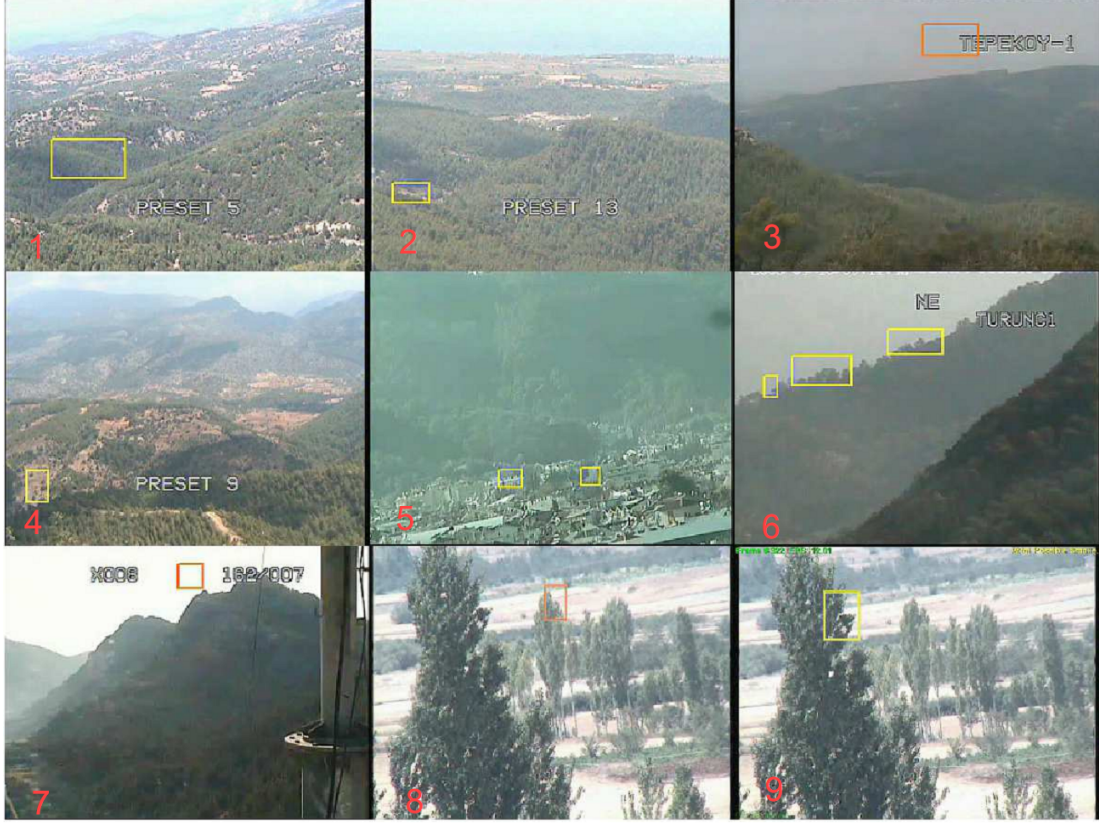


Figure 2.3: False alarms issued to videos from Table 2.3. The first two and the last two images are from the same video sequences. Cloud shadows, clouds, fog, moving tree leaves, and sunlight reflecting from buildings cause false alarms

Table 2.3: Eight different algorithms are compared in terms of false alarm rates.

Video	Frames	False Alarm Rates							
		FIXED	ULP	NLMS	NLMS-B	ECF	ECF-B	LOGX	LOG(X+1)
V11	6300	0.03%	0.03%	0.03%	0.03%	0.02%	0.03%	0.03%	0.03%
V12	3370	7.00%	2.97%	1.01%	1.96%	0.92%	1.01%	1.66%	0.89%
V13	7500	3.13%	3.12%	2.77%	2.77%	2.77%	2.77%	2.24%	2.77%
V14	6294	17.25%	9.64%	2.27%	2.67%	2.18%	2.40%	3.23%	4.89%
V15	6100	4.33%	4.21%	2.72%	2.75%	1.80%	2.75%	1.23%	2.97%
V16	433	11.32%	11.32%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
V17	7500	0.99%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Average	-	6.29%	4.47%	1.26%	1.46%	1.10%	1.28%	1.20%	1.65%

Figure 2.3 shows typical false alarms issued to videos by non-adaptive methods. Figure 2.4 shows the squared pixels errors of NLMS and ECF based schemes are

compared for the video clip *V12*. The average pixel error for a video sequence v is calculated as follows:

$$\bar{E}(v) = \frac{1}{F_P} \sum_{n=1}^{F_P} \left(\frac{e_n}{N_P} \right), \quad (2.21)$$

where N_P is the number of pixels, F_P is the number of frames in the video sequence, and e_n is the sum of the squared errors for each classified pixel in image frame n . The figure shows the average errors for the frames between 500 and 900 of *V12*. Between frames 510 and 800, the camera moves to a new position and weights are reset to their initial values. Since the tests are performed with offline videos we do not know the location of the camera, therefore we do not keep separate weights for each preset. Instead we reset the algorithm when the camera moves. The ECF algorithm achieves convergence faster than the NLMS algorithm. The tracking performance of the ECF algorithm is also better than the NLMS based algorithm which can be observed after the frame number 600, at which point some of the weak classifiers issue false alarms.

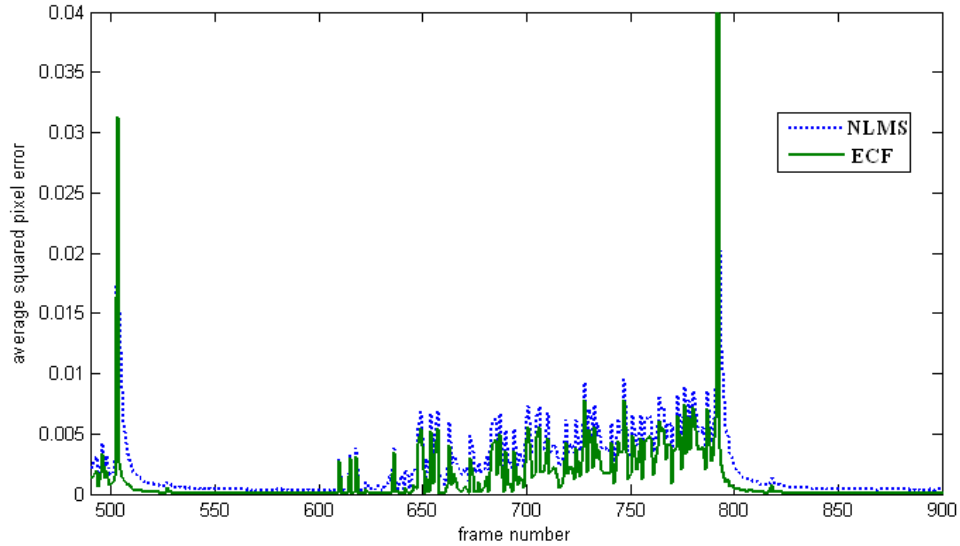
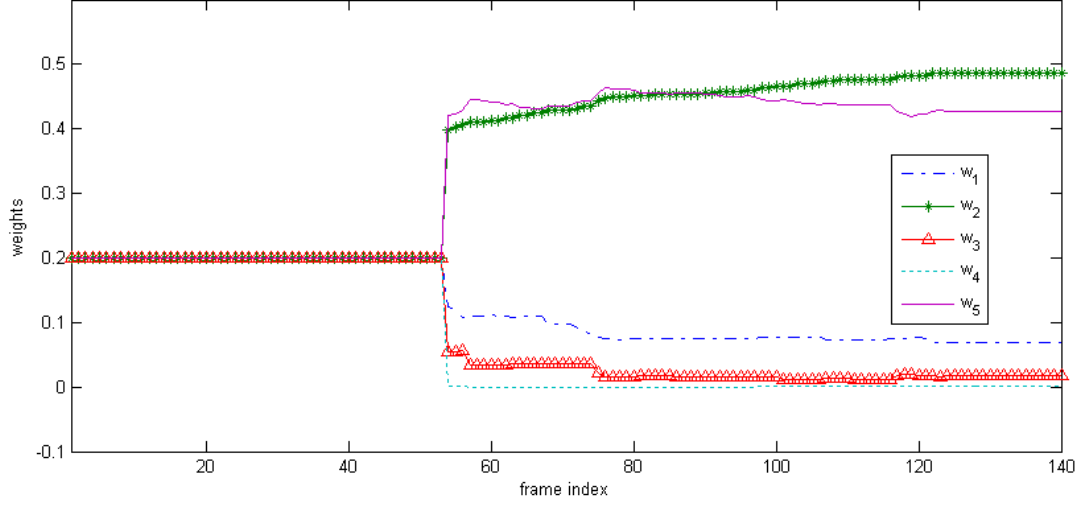


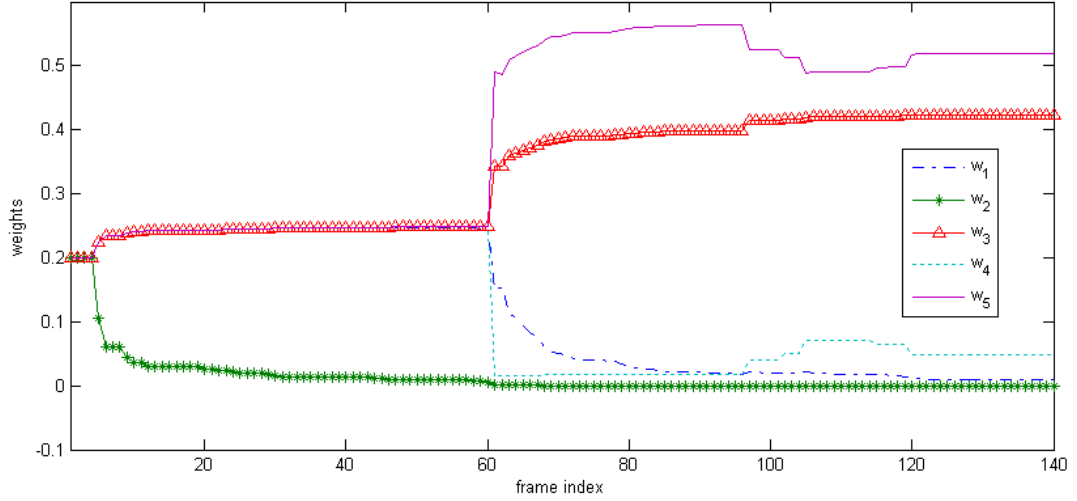
Figure 2.4: Average squared pixel errors for the NLMS and the ECF based algorithms for the video sequence *V12*.

Figure 2.5 depicts the weights of two different pixels from *V12* for 140 frames. For the first pixel, $x_t(1)$, $x_t(3)$ and $x_t(4)$ get closer to 1 after the 60th frame,

and therefore, their weights are reduced. For the second pixel, $x_t(2)$ issues false alarms after the 4th frame; $x_t(2)$ and $x_t(4)$ issue false alarms after the 60th frame.



(a) Adaptation of weights for a pixel at $x = (55, 86)$ in V12.



(b) Adaptation of weights for a pixel at $x = (56, 85)$ in V12.

Figure 2.5: Adaptation of weights in a video that do not contain smoke.

Chapter 3

Generalized Update Equations for Online Adaptive Learning

3.1 Introduction

Online learning algorithms have wide range of applications some of which include speech recognition [47], financial prediction [48], and image processing [49, 50]. In the online learning framework, the algorithm adjusts its parameters using sequentially received samples and according user feedback that determines the correct classification result. One of the earlier online learning algorithms was the perceptron [51]. In online binary classification application, perceptron finds a separating hyperplane between two classes. The coefficients of the hyperplane are called the weights of the algorithm. Perceptron updates the weights when there is misclassification of the incoming data [52]. Passive-Aggressive (PA) online learning algorithms solve an optimization that represent a trade-off between adjusting the weights according to new data and retaining the old weights [15]. PA algorithms are margin based methods that try to increase the margin between the samples and the supporting hyperplane [15]. There are many similar algorithms based on maximum margin classification idea in machine learning literature [53–56].

Previously discussed algorithms work on linearly separable data. When the classes are not linearly separable, specific kernels map the data to higher dimensional space. The so-called kernel trick has been successfully used especially in support vector machine classification algorithms [57]. Kernelized versions of

perceptron and other online learning algorithms generalize the method for non-linearly separable classification tasks [58]. The most important problem with kernel based algorithms is that they require the storage of every incorrectly classified sample as a support vector. This quickly increases the memory requirement of the algorithm [59]. There are many publications that suggest solutions for this problem. Some methods remove the previous support vectors, some employ more sophisticated strategies to decide which support vectors to keep as their number exceeds a threshold [60–63].

In this work, we investigate maximum-margin online learning algorithms and we propose a real-time detection problem that can be solved using the developed methods. We analyze kernelized versions of the algorithms. We use the method described in [59] to keep the number of support vectors bounded for kernel algorithms. We verify the performance of the algorithms with extensive experimentation. We also propose a novel solution to a real-time computer vision problem using the described algorithms. The real-time problem is wildfire detection from aerial image sequences which is not implemented before. There are many wildfire detection applications that work with stationary or PTZ (pan-tilt-zoom) cameras mounted on forest watch towers [8, 9]. We propose a new method that can be used with cameras mounted on aerial platforms.

The rest of the chapter is organized as follows. In Section 3.2, we introduce the notation and studied online learning algorithms. In Section 3.3, we formulate the algorithms, obtain their kernel versions. Section 3.4 explains the wildfire detection algorithm based on online learning, the experimental results are presented in Section 3.5.

3.2 Online Learning Review

In this chapter, boldface lowercase letters represent column vectors and bold uppercase letters represent matrices. Let $\mathbf{v}$ be a vector, $\|\mathbf{v}\|$ denotes the norm, $\mathbf{v}(i)$ is the i_{th} element, and $\mathbf{v}^T$ denotes the transpose.

We consider online classification. In binary classification at each time iteration t , a feature vector $\mathbf{x}_t$, with label y_t is observed. As the classification algorithm we use a linear classifier in which the output is given by $\text{sign}(\mathbf{w}_t^T \mathbf{x}_t)$. The aim of

the algorithm is to learn the weights of a separating hyperplane, $\mathbf{w}_t$. We update the weight vector $\mathbf{w}_t$ iteratively using an update function that depends on the optimization problem. In online passive-aggressive algorithms [15] the goal is try to maximize the margin defined by $y_t(\mathbf{w}_t^T \mathbf{x}_t)$. The basic optimization problem solved by this algorithm is given as

$$\mathbf{w}_{t+1} = \arg \min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w} - \mathbf{w}_t\|^2 \quad \text{subject to} \quad L(\mathbf{w}; (\mathbf{x}_t, y_t)) = 0, \quad (3.1)$$

where $L(\mathbf{w}; (\mathbf{x}_t, y_t)) = \max(0, 1 - y_t(\mathbf{w}^T \mathbf{x}_t))$ is the hinge loss function which is 0 when the classification is correct and the algorithm *passively* retains the old weight vector. When the classification is incorrect the algorithm tries to achieve the margin value of 1 by *aggressively* modifying the weight vector. Such aggressive weight updates can cause problems in noisy environments. PA-I and PA-II algorithms use smoother update functions and they can still work in noisy environments. The optimization problem for the PA-I algorithm is as follows:

$$\mathbf{w}_{t+1} = \arg \min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w} - \mathbf{w}_t\|^2 + C\xi \quad \text{s. t.} \quad L(\mathbf{w}; (\mathbf{x}_t, y_t)) < \xi \quad \text{and} \quad \xi \geq 0. \quad (3.2)$$

The update equation obtained by solving 3.2 is

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \min \left\{ C, \frac{L_t}{\|\mathbf{x}_t\|^2} \right\} y_t \mathbf{x}_t. \quad (3.3)$$

Similarly, the optimization problem for the PA-II algorithm is as follows

$$\mathbf{w}_{t+1} = \arg \min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w} - \mathbf{w}_t\|^2 + C\xi^2 \quad \text{s. t.} \quad L(\mathbf{w}; (\mathbf{x}_t, y_t)) < \xi, \quad (3.4)$$

The corresponding update equation is given by

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \frac{L_t}{\|\mathbf{x}_t\|^2 + \frac{1}{2C}} y_t \mathbf{x}_t. \quad (3.5)$$

where C is the *aggressive parameter* of the algorithms, and $L_t = \max(0, 1 - y_t(\mathbf{w}_t^T \mathbf{x}_t))$.

In the next section, we develop online learning algorithms from a generic optimization problem.

3.3 Online Learning Algorithms

We use the following generic update equation to develop maximum margin online learning algorithms

$$\mathbf{w}_{t+1} = \arg \min_{\mathbf{w} \in \mathcal{S}} \{D(\mathbf{w}, \mathbf{w}_t) + \mu L(\mathbf{w}; (\mathbf{x}_t, y_t))\},$$

where $\mathcal{S}$ is the set of the parameters, $D(\mathbf{w}, \mathbf{w}_t)$ is the distance function, and $L(\mathbf{w}; (\mathbf{x}_t, d_t))$ is the loss, $\mathbf{w}_t$ denotes the weight vector and y_t is the class label. We use different divergence measures and loss functions to obtain new update methods which might be more suitable for specific applications. We propose on-line learning algorithms that try to minimize the squared loss instead of imposing a strict hyperplane bound on the optimization. We derive the first algorithm from the following optimization

$$\mathbf{w}_{t+1} = \arg \min_{\mathbf{w}} \left\{ \|\mathbf{w} - \mathbf{w}_t\|_{\mathbf{R}_t}^2 + \mu L(\mathbf{w}; (\mathbf{x}_t, y_t))^2 \right\}. \quad (3.6)$$

In this method we control the passive-aggressiveness by the update parameter μ . The update equation corresponding to this algorithm is

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \mu L_t y_t \frac{\mathbf{R}_t^{-1} \mathbf{x}_t}{1 + \mu \mathbf{x}_t^T \mathbf{R}_t^{-1} \mathbf{x}_t}, \quad (3.7)$$

where $\mathbf{R}_t$ is related to the autocorrelation matrix of the data and can be estimated as $\mathbf{R}_t = \beta \mathbf{R}_{t-1} + \alpha \mathbf{x}_t \mathbf{x}_t^T$. We name this algorithm OL-RLS, which stands for online learning recursive least squares. This method becomes similar to PA-II algorithm when $\mathbf{R}_t$ is chosen as the identity matrix. This method is related to Newton-Raphson optimization method in its use of the higher order statistics of the input samples.

It is possible to simplify the update equation by using the Taylor series approximation of the loss function around the current weight vector $\mathbf{w}_t$. Then, the loss function becomes

$$L(\mathbf{w}; (\mathbf{x}_t, y_t)) = L_t^2 - 2L_t y_t \mathbf{x}_t^T (\mathbf{w} - \mathbf{w}_t) + O(\|\mathbf{w}\|^2). \quad (3.8)$$

Using this approximation and ignoring higher order terms the update equation reduces to

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \mu \mathbf{R}_t^{-1} L_t y_t \mathbf{x}_t. \quad (3.9)$$

We obtain NLMS (Normalized LMS) based algorithm from this update equation by using $\mathbf{R}_t = \mathbf{x}_t^T \mathbf{x}_t \mathbf{I}$. We name this algorithm OL-LMS whose update equation becomes

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \mu \frac{L_t y_t \mathbf{x}_t}{\mathbf{x}_t^T \mathbf{x}_t}. \quad (3.10)$$

OL-LMS algorithm is also related to stochastic gradient methods since it only uses the current sample to update the weight vector.

We derive other update methods using Bregman divergences as the distance measures [37]

$$D(\mathbf{w}, \mathbf{w}_t) = f(\mathbf{w}) - f(\mathbf{w}_t) - \nabla f(\mathbf{w}_t)^T(\mathbf{w} - \mathbf{w}_t), \quad (3.11)$$

where $f(\mathbf{w})$ is a convex distance function. When $f(\mathbf{w}) = \|\mathbf{w}\|^2$ the Bregman divergence reduces to the Euclidean distance. Using Bregman divergence and Taylor series approximation of the squared loss function, we solve the optimization problem using the following simplified formula

$$\nabla f(\mathbf{w}_{t+1}) = \nabla f(\mathbf{w}_t) - \mu \nabla L(\mathbf{w}_t; (\mathbf{x}_t, y_t)). \quad (3.12)$$

The most common convex functions used in Bregman divergences are, l_2 norm $f(\mathbf{w}) = \|\mathbf{w}\|^2$, Shannon entropy $f(\mathbf{w}) = \sum_i \mathbf{w}(i) \log(\mathbf{w}(i))$ and Burg's entropy $f(\mathbf{w}) = -\sum_i \log(\mathbf{w}(i))$ [36]. When we use $f(\mathbf{w}) = \sum_i \mathbf{w}(i) \log(\mathbf{w}(i))$ cost function and the Taylor series approximation of the squared loss function we obtain the following update functions

$$1 + \log(\mathbf{w}_{t+1}) = 1 + \log(\mathbf{w}_t) + 2\mu y_t L_t \mathbf{x}_t.$$

After rearranging we get

$$\mathbf{w}_{t+1} = \mathbf{w}_t \exp(\mu y_t L_t \mathbf{x}_t).$$

This update equation is valid for non-negative weights. We include negative weights by expressing the weight vector in the following form

$$\mathbf{w}_t = \mathbf{w}\mathbf{p}_t - \mathbf{w}\mathbf{n}_t,$$

where $\mathbf{w}\mathbf{p}$ and $\mathbf{w}\mathbf{n}$ contain positive valued weights, respectively. These are updated using

$$\mathbf{w}\mathbf{p}_{t+1} = \mathbf{w}\mathbf{p}_t \exp(\mu y_t L_t \mathbf{x}_t),$$

and

$$\mathbf{w}\mathbf{n}_{t+1} = \mathbf{w}\mathbf{n}_t \exp(-\mu y_t L_t \mathbf{x}_t),$$

respectively.

This algorithm is also called exponentiated gradient (EG) in the literature. Therefore, we name the Bregman divergence based algorithm OL-EG. Recently, entropy based distance measures have been successfully used in adaptive filtering and

classifier fusion applications [9, 18]. However, there is one big disadvantage for entropy based algorithms in online learning applications: they cannot easily be generalized with a kernel operator.

3.3.1 Kernel Versions of Adaptive Algorithms

Euclidean distance based algorithms satisfy the following equation when the initial weights are set to zero

$$\mathbf{w}_t = \sum_{i=1}^{t-1} c_t y_t \mathbf{x}_i, \quad (3.13)$$

where c_t is the update parameter that changes depending on the particular algorithm being used. Then, the product that will input to the sign classification function is

$$\mathbf{w}_t^T \mathbf{x}_t = \sum_{i=1}^{t-1} c_t y_t (\mathbf{x}_i^T \mathbf{x}_t). \quad (3.14)$$

We can replace the dot product in the above equation with a Mercer kernel as $\kappa(\mathbf{x}_i, \mathbf{x}_t)$ to enable the classification of non-separable classes [64].

OL-LMS algorithm can be easily kernelized in this way. OL-RLS algorithm's kernel application is more involved than the LMS based algorithms [16]. We can write the Kernel version of OL-RLS algorithm in the following form:

$$\begin{aligned} &\text{Initialization} \\ &\quad \mathbf{Q}_1 = (\mu + \kappa(\mathbf{x}_1, \mathbf{x}_1))^{-1} \\ &\quad \mathbf{w}_1 = \mathbf{Q}_1 \\ &\text{Iterations} \\ &\quad \mathbf{h}_t = [\kappa(\mathbf{x}_t, \mathbf{x}_1), \dots, \kappa(\mathbf{x}_t, \mathbf{x}_{t-1})]^T \\ &\quad \mathbf{z}_t = \mathbf{Q}_{t-1} \mathbf{h}_t \\ &\quad r_t = \mu + \kappa(\mathbf{x}_t, \mathbf{x}_t) - \mathbf{z}_t^T \mathbf{h}_t \\ &\quad \mathbf{Q}_t = r_t^{-1} \begin{bmatrix} \mathbf{Q}_{t-1} r_t + \mathbf{z}_t \mathbf{z}_t^T & -\mathbf{z}_t \\ -\mathbf{z}_t^T & 1 \end{bmatrix} \\ &\quad L_t = 1 - \mathbf{h}_t^T \mathbf{a}_{t-1} \\ &\quad \mathbf{w}_t = \begin{bmatrix} \mathbf{w}_{t-1} - \mathbf{z}_t L_t / r_t \\ -L_t / r_t \end{bmatrix}. \end{aligned} \quad (3.15)$$

We cannot write OL-EG algorithm in its current form using kernels. We use the modified version of the entropic update formula [65] that allows the use of kernels. The method is based on the online mirror descent (OMD) algorithm [66]. OMD tries to generalize the online learning algorithms using tools from convex

optimization theory. We can write the modified entropic update algorithm [65] as follows

$$\begin{aligned}
&\text{Initialization} && 0.882 < a < 1.109, \eta > 0, \delta > 0 \\
&\theta_1 &= & 0 \\
&H_0 &= & \delta \\
&\text{Iterations} \\
&H_t &= & H_{t-1} + \eta^2 \max(\|x_t\|, \|x_t\|^2) \\
&\alpha_t &= & a\sqrt{H_t} \\
&\beta_t &= & H_t^{3/2} \\
&\mathbf{w}_t &= & \frac{\theta_t}{\beta_t \|\theta_t\|} \exp\left(\frac{\|\theta_t\|}{\alpha_t}\right) \\
&\theta_{t+1} &= & \theta_t - y_t L_t \mathbf{x}_t
\end{aligned} \tag{3.16}$$

The above formulation only depends on the inner products between the vectors and therefore can be easily written in kernel form.

3.3.2 Mixture of Online Learners

In the online learning algorithms presented above an important consideration is the selection of the algorithm and the update parameter for a specific application. Update parameter determines both the convergence rate and the steady-state error-rate of the algorithm. The selection of the classifier and dependence on the update parameter can be somewhat relieved by using ideas from the theory of classifier ensembles. This framework is also called classifier fusion or mixture of experts [24]. It might be possible to combine the outputs of different learners using weights that are trained to minimize the overall loss of the algorithm. We can obtain different learners from online learners with different update parameters or different update formulations. We can use the formulation from adaptive filtering framework to adjust the weights assigned to different learners. Figure 3.1 shows the general scheme of learner combination methods.

Let us assume the output of the i_{th} online learner is denoted by

$$z_t(i) = \mathbf{w}_t^T \mathbf{x}_t \quad i = 1, \dots, M. \tag{3.17}$$

We determine the weights of the combiner, ψ , by solving the following optimization problem

$$\psi_{t+1} = \arg \min_{\psi} \{D(\psi, \psi_t) + \mu L(\psi; (\mathbf{z}_t, y_t))\} + \lambda \left(\sum_i \psi(i) - 1 \right). \tag{3.18}$$

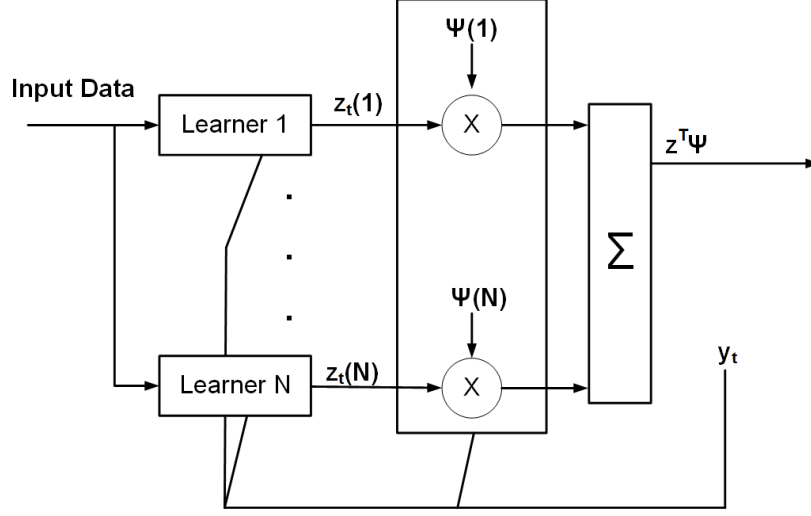


Figure 3.1: Fusion of Online Learners

The combiner tries to reduce the total loss of the system by adjusting the weights. We add the last constraint in the above formulation to ensure that the sum of the weights equals to 1, i.e., $\sum_{i=1}^N \psi(i) = 1$. We also impose non-negativity constraint on the weights, $\psi(i) > 0$. We can satisfy both constraints if we use Bregman divergence with Shannon entropy as the distance measure. We obtain the solution of the optimization as [17]

$$\psi_{t+1}(i) = \frac{\psi_t(i) \exp(\mu y_t L_t \mathbf{z}_t(i))}{\sum_{j=1}^M \psi_t(j) \exp(\mu y_t L_t \mathbf{z}_t(j))} \quad i = 1, \dots, N. \quad (3.19)$$

The fusion of different learners proves useful when the values of the update parameter cannot be estimated with confidence and in this case the mixture algorithm works by increasing the success rate of many weak learners using linear weighting.

3.4 Wildfire Detection with Online Classifiers

There are many wildfire detection applications that work with stationary or PTZ cameras mounted on forest surveillance towers [8, 9]. The coverage area of these systems is limited by the location of the tower and field of view of the camera. Larger areas can be covered using PTZ cameras in scan mode but this may increase the average response time of the detection algorithms. Therefore, we propose to use cameras mounted on unmanned aerial vehicles (UAVs) or small

aircrafts for early detection of wildfires.

Online learning algorithms are a good candidate for this problem, since the UAV is always in motion and environmental conditions and the camera's field of view changes constantly. We can stream the image sequences obtained from the moving vehicle to the watch tower for analysis and feedback and the detection algorithm can adjust its parameters accordingly.

Wildfire detection algorithms in the literature try to detect smoke which starts before the fire for most wildfire scenarios. Most algorithms first isolate the slow moving regions in stationary camera image sequences which is a good indication of wildfire smoke. However, for the case of aerial image sequences obtained from moving platforms the movement of the smoke is not clearly discernible. Therefore, in this work, we propose a different method for wildfire smoke detection without using motion information. We first perform a global segmentation of a single image using a method that can differentiate smoke regions as a separate segment. Then we extract features from candidate segments and classify these features using online learning algorithms. Figure 3.2 shows the main steps of the wildfire detection algorithm.

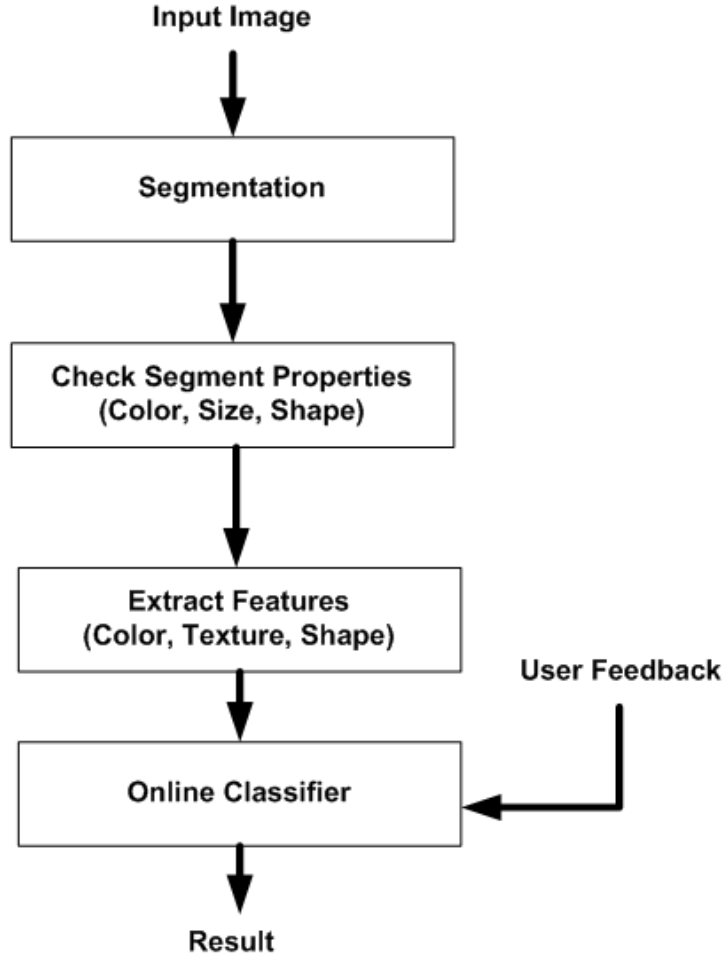


Figure 3.2: Wildfire detection algorithm flowchart.

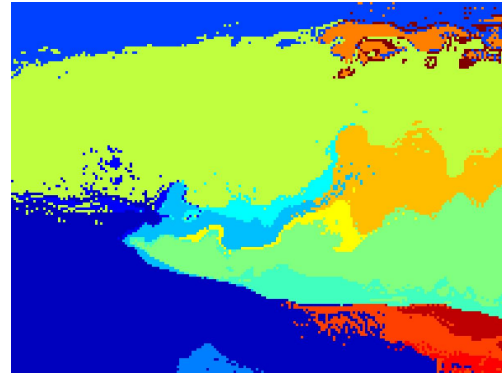
3.4.1 Image Segmentation

We use a graph-based method for image segmentation [67]. In graph based segmentation algorithms each pixel corresponds to a node in the graph and some of the pixels are connected by edges. Weights of edges measure the degree of contrast between the pixels and segmentation works by selecting edges from the graph. The method in [67] adjusts the segmentation criteria using the difference between neighboring segments of the image. Segmentation works by selecting edges from the graph. This method uses pairwise region comparison for segmentation. The computational cost of the method is $m \log(m)$ for m graph edges and the algorithm takes less than 1 seconds for 240×320 images. This algorithm is

very successful in maintaining detail in low-pass regions and ignoring detail in high-pass regions. Therefore, the method can segment smoke regions which usually have low-frequency characteristics compared to the environment and it can also distinguish non-smoke regions with irregular textures as a whole segment. In Figure 3.3 sample segmentation results are displayed for actual aerial wildfire smoke images. We see that the smoke regions are successfully segmented for all images.



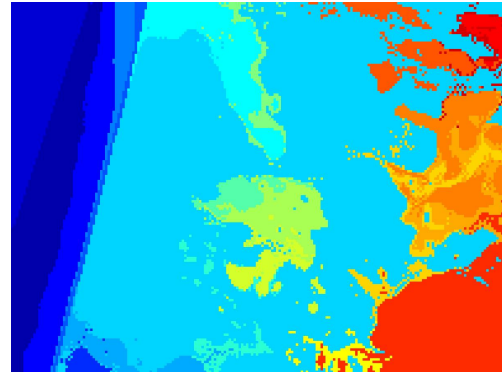
(a) Visible Image



(b) Segmentation



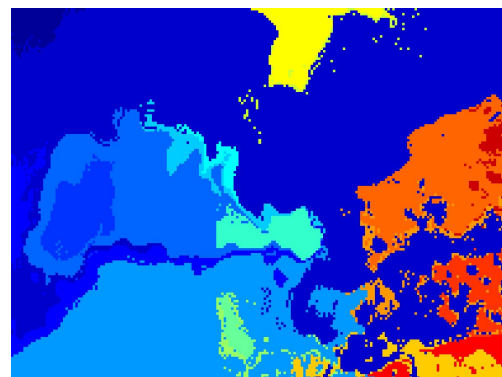
(c) Visible Image



(d) Segmentation



(e) Visible Image



(f) Segmentation

Figure 3.3: Example segmentation results for actual wildfire smoke images.

3.4.2 Color Analysis

After segmentation we perform color analysis of detected segments to determine candidate smoke regions. Wildfire smoke usually has a gray-to-white color that results from burning vegetation. We use thresholds in YUV color space to determine color content of segments. The color condition

$$M_C = Y > T_Y \quad \& \quad |U - 128| < T_U \quad \& \quad |V - 128| < T_V \quad (3.20)$$

where Y , U , V denote separate channels of the color space, T_Y , T_U and T_V are predetermined thresholds and M_C is the color mask for the whole image. For each segment if the ratio of the pixels satisfying the color condition is above a threshold, this segment is further processed for feature extraction and classification.

3.4.3 Feature Extraction

We extract color, texture and shape features to represent distinct characteristics of regions. We use 3D histograms in RGB color space to extract color features. We also add mean values of R,G,B and Y,U,V channels and entropy of the histogram as additional features. We use 8 bins for each channel and therefore the size of the color feature vector is $512+7=519$.

We use the dual-tree complex wavelet transform (DT-CWT) to extract texture features. Dual-tree complex wavelet transform tree, is recently developed to overcome the shortcomings of conventional wavelet transform, such as shift variance and poor directional selectivity [68]. In DT-CWT there are six directional subbands for each decomposition level. We calculate the mean, variance, kurtosis, and skewness of the absolute values of the complex wavelet coefficients corresponding to each subband at each level. Therefore for a two-level transform we obtain 48 element feature vectors for texture modeling.

We use Zernike moments to extract shape features from images. Zernike moment of order n and repetition m for an $K \times K$ image, I , can be defined as follows [69]

$$Z_{n,m} = \frac{n+1}{\gamma_N} \sum_{c=0}^{K-1} \sum_{r=0}^{K-1} I(c,r) R_{n,m}(\rho_{cr}) \exp(-jm\theta_{cr}), \quad (3.21)$$

where $R_{n,m}$ is the Zernike radial polynomial

$$R_{n,m} = \sum_{s=0}^{(n-|m|)/2} (-1)^s \frac{(n-s)!}{s!((n+|m|)/2-s)!((n-|m|)/2-s)!} \rho^{n-2s}, \quad (3.22)$$

ρ_{rc} is the distance and θ_{rc} is the angle of the point (r, c) with respect to the origin when the image is mapped to the unit circle, γ_N is a normalization factor. Similar to [69] we extract 32-element feature vectors for different n and m values which should satisfy, $n - |m| = \text{even}$ and $|m| \leq n$.

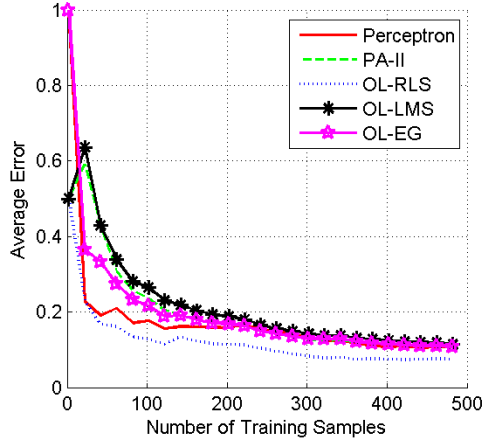
3.5 Experimental Results

In the experiments we compare the formulated methods against perceptron and PA-II based algorithms both on synthetic and real datasets.

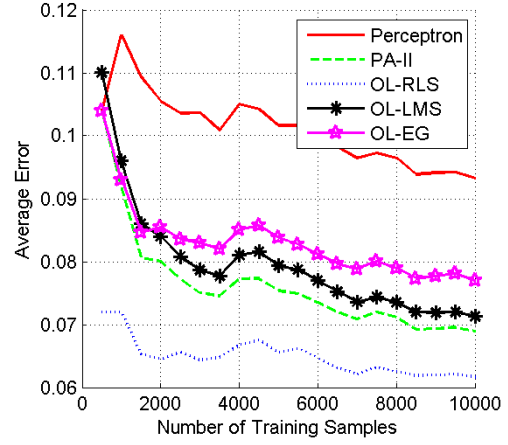
3.5.1 Experiments on Synthetic and Standard Datasets

In the first experiment we compare the developed online learning algorithms in classifying linearly separable classes created from standard normal distribution. We use DOGMA toolbox [70] implementations of competing algorithms in the experiments. For the OL-RLS algorithm we set the parameters of the autocorrelation function $\mathbf{R}_t = \beta \mathbf{R}_{t-1} + \alpha \mathbf{x}_t \mathbf{x}_t^T$ to $\beta = 1$ and $\alpha = 0.2$.

In Figure 3.4 average error rates for different algorithms are displayed on a synthetic dataset without instance or label noise. In this example OL-RLS algorithm performs better than perceptron, PA-II and other algorithms. It has a faster convergence rate and steady-state error value than other algorithms. Other algorithms have similar performance to PA-II and as expected simple perceptron has the worst results.

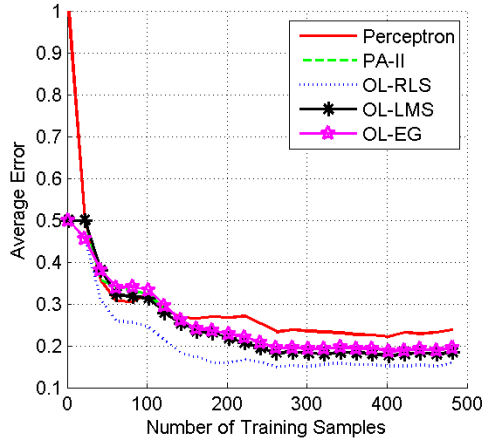


(a) First 500 samples

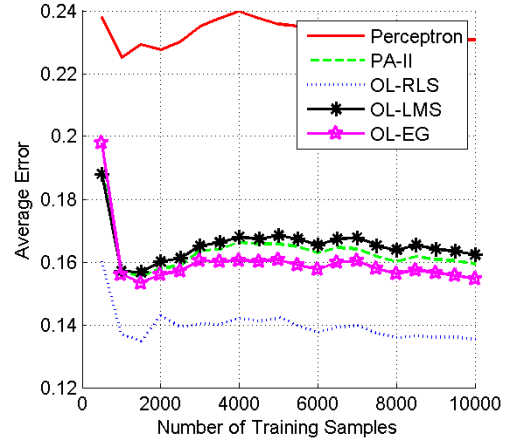


(b) Rest of the samples.

Figure 3.4: Average errors for different online learning algorithms for a synthetic dataset without noise.



(a) First 500 samples



(b) Rest of the samples.

Figure 3.5: Average errors for different online learning algorithms for a synthetic dataset with 8% label noise and instance noise ($\sigma_n = 0.1$).

In Figure 3.5, we compare the same algorithms when there is 8% label noise and also instance noise with parameter $\sigma_n = 0.1$ in the samples. When we add noise to the system, the performance gap between the new methods (OL-RLS and OL-EG especially) and other algorithms becomes more noticeable.

Figure 3.6 shows the average error rates of OL-LMS algorithm for different values of the update parameter μ and also shows the error rate of the combined learner (denoted as OL-M) that we obtain by fusing the results of five different learners. As expected when we increase μ the initial convergence is faster but the eventual error rate increases. The error rate of the combined algorithm is less than all of the algorithms in the mixture. In Figure 3.7 we perform the same test for larger values of the update parameter. In this case the first algorithm already achieves the minimum possible error rate and the combination result tries to track this solution.

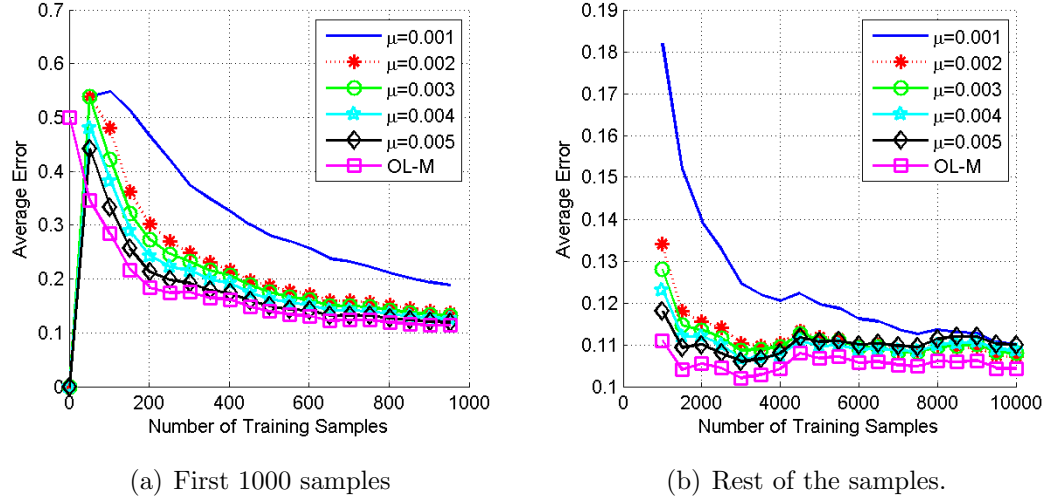


Figure 3.6: Average errors of OL-LMS algorithm for different values of the update parameter, μ and the result of the learner combination.

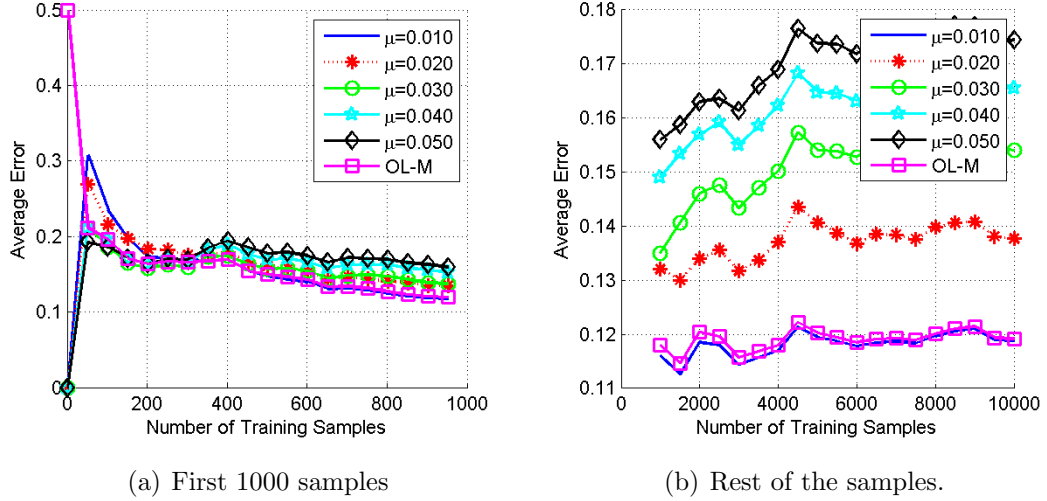


Figure 3.7: Average errors of OL-LMS algorithm for different values of the update parameter, μ and the result of the learner combination.

In the second experiment we test the kernel versions of the algorithms with standard machine learning datasets from the UCI (University of California, Irvine) machine learning repository [71]. We use half of the samples for training and the other rest for testing. We use a Gaussian kernel, $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2)$, with $\gamma = 0.9$. In Table 3.1 we give the descriptions of datasets used in the experiments. Figure 3.8 displays average error rates of kernel based algorithms on the D01 dataset. We form the OL-M algorithm from the mixture of five OL-LMS algorithms with different update rates. Generally OL-RLS algorithm show better performance on this dataset. Table 3.2 summarizes the average success rates of the algorithms on training set (R1) and also shows the success rate of final classifier on the test set (R2).

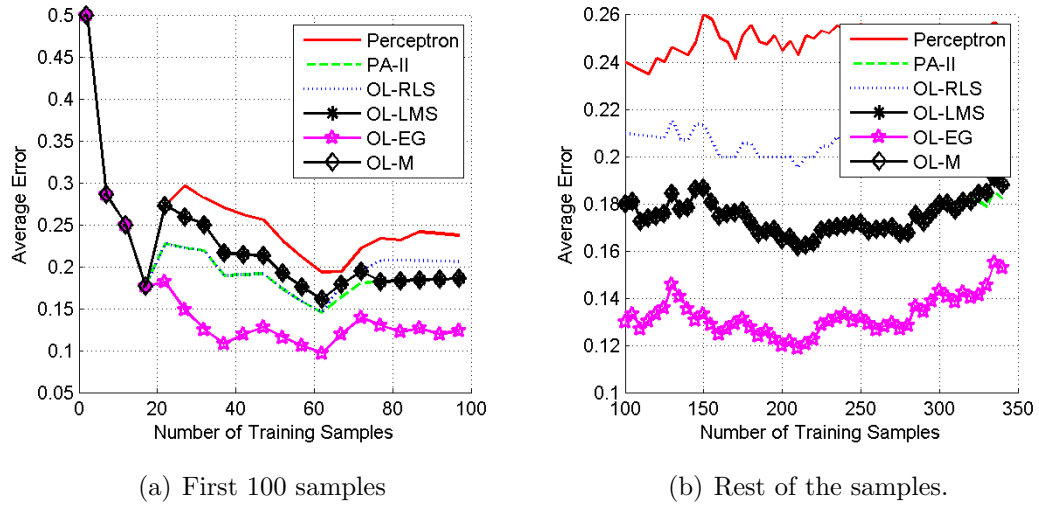


Figure 3.8: Average errors of kernel based algorithms on D01 dataset.

Table 3.1: Descriptions of datasets used in kernel experiments.

Dataset	Name	# of Samples	# of Features
D01	Australian (Statlog)	690	14
D02	Wisconsin B. Cancer	683	10
D03	Heart (Statlog)	270	13
D04	Ionosphere	351	34
D05	Sonar	208	60
D06	Splice	1000	60
D07	Adult	2265	123

Table 3.2: Success rates of kernel based algorithms on UCI machine learning datasets.

		Perceptron	PA-II	OL-RLS	OL-LMS	OL-EG	OL-M
D01	R1	75.63%	82.55%	79.63%	82.36%	86.91%	82.36%
	R2	74.13%	86.92%	86.05%	86.63%	86.34%	86.63%
D02	R1	91.72%	94.33%	93.65%	94.73%	94.49%	94.73%
	R2	95.89%	97.65%	97.65%	97.65%	97.65%	97.65%
D03	R1	69.67%	70.99%	70.19%	71.85%	75.24%	71.85%
	R2	74.81%	79.26%	78.52%	79.26%	80.74%	79.26%
D04	R1	78.77%	81.40%	83.88%	81.40%	80.22%	81.40%
	R2	76.57%	85.71%	89.14%	85.71%	87.43%	85.71%
D05	R1	69.93%	76.39%	76.39%	76.30%	60.91%	76.30%
	R2	76.70%	81.55%	84.47%	81.55%	62.14%	81.55%
D06	R1	63.37%	63.57%	63.57%	63.57%	62.09%	63.57%
	R2	65.93%	66.33%	66.33%	66.33%	65.73%	66.33%
D07	R1	70.41%	75.77%	75.88%	75.80%	63.17%	75.83%
	R2	75.88%	81.45%	81.18%	81.54%	69.96%	81.45%

In the final experiment we test the real-time performance of the algorithms on the pedestrian detection problem from thermal image sequences. We use the dataset generated in [72]. The dataset has 10 short videos that depict various pedestrians mostly in motion. We used region covariance descriptor to extract features from the videos [73]. For each sequence we use first five frames as training frames and provide full feedback to the algorithm. After the training stage we only provide feedback at sparse intervals of 20 frames. We use the positions of the pedestrians to extract the positive samples and random locations in images to extract the negative samples. In Figure 3.10 we show sample images of positive and negative samples. Notice the incorrectly classified background samples in Vid05. Figure 3.9 shows the average error rates of the algorithms on Vid10 sequence.

Table 3.3 shows the success rates of algorithms. We determine the success rate as the ratio of correctly classified samples to the number of all samples provided

to the algorithms. From the results we see that OL-M mixture algorithm has the best overall performance. Another inference is the weak performance of the OL-EG algorithm.

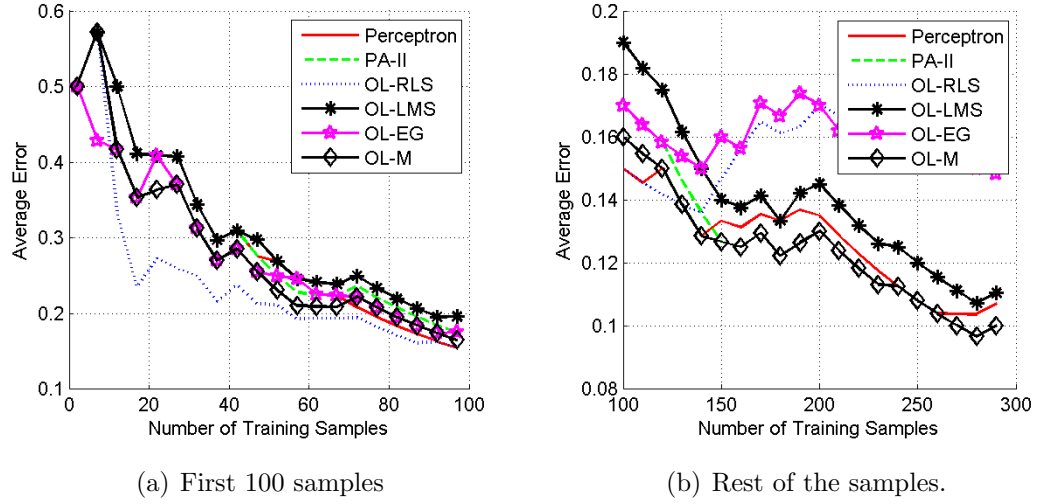


Figure 3.9: Average errors of kernel based algorithms on Vid10 sequence.



Figure 3.10: Sample image frames from video sequences. Red rectangles mark the pedestrians, green rectangles mark the background samples.

Table 3.3: Success rates of kernel based algorithms on thermal pedestrian dataset.

	Perceptron	PA-II	OL-RLS	OL-LMS	OL-EG	OL-M
Vid01	0.95%	0.97%	0.91%	0.95%	0.82%	0.96%
Vid02	0.96%	0.96%	0.97%	0.97%	0.81%	0.97%
Vid03	0.76%	0.87%	0.82%	0.84%	0.60%	0.87%
Vid04	0.95%	0.96%	0.94%	0.96%	0.76%	0.96%
Vid05	0.89%	0.90%	0.87%	0.90%	0.78%	0.91%
Vid06	0.97%	0.98%	0.98%	0.98%	0.83%	0.98%
Vid07	0.97%	0.97%	0.95%	0.95%	0.83%	0.97%
Vid08	0.97%	0.96%	0.95%	0.97%	0.74%	0.96%
Vid09	0.99%	0.98%	0.99%	0.98%	0.90%	0.98%
Vid10	0.92%	0.94%	0.84%	0.93%	0.87%	0.94%

3.5.2 Wildfire Detection Experiments

We use actual wildfire footage recorded from helicopters. We use first five frames of videos for training and after that we only provide feedback every 20 frames. The algorithm can process a frame at approximately 1 seconds. Therefore in the experiment feedback is provided every 20 seconds. Table 3.4 summarizes the results of the wildfire detection experiments. We compose the first video from the frames accumulated from other videos so that it can give an idea about the generalization performance of the classifiers. The table displays the success rate (SR) and the number of support vectors (SV) for each classifier. Success rate is calculated as the percentage of correctly classified samples to the total number of samples. The number of frames and samples in each video is also provided. From the results we see that all adaptive learning algorithms show similar performance in classifying wildfire smoke images. This hints that the developed features are optimal in the sense that they can be separated with any maximum-margin classifier.

Table 3.4: Test results on wildfire detection experiment.

		Perceptron	PA-II	OL-RLS	OL-LMS	OL-EG	OL-M
Vid01 Frames: 42 Samples: 429	SR	90.91%	90.91%	90.91%	90.91%	90.44%	90.91%
	SV	63	63	63	63	63	315
Vid02 Frames: 106 Samples: 769	SR	87.00%	87.00%	87.00%	87.00%	86.74%	87.00%
	SV	93	93	93	93	93	465
Vid03 Frames: 568 Samples: 4738	SR	91.39%	90.84%	92.09%	90.61%	90.42%	91.83%
	SV	247	344	343	344	339	1715
Vid04 Frames: 735 Samples: 6001	SR	92.95%	92.98%	93.13%	92.95%	93.53%	93.05%
	SV	326	344	344	344	344	1720
Vid05 Frames: 902 Samples: 9294	SR	94.53%	94.43%	94.65%	94.34%	91.94%	94.47%
	SV	534	619	613	619	619	3040
Vid06 Frames: 1143 Samples: 8924	SR	92.74%	92.72%	92.62%	92.69%	92.59%	92.78%
	SV	471	507	507	507	507	2535
Vid07 Frames: 631 Samples: 6659	SR	92.58%	92.48%	92.49%	92.48%	92.18%	92.51%
	SV	294	390	384	390	390	1925
Vid08 Frames: 746 Samples: 8185	SR	91.40%	91.29%	91.83%	91.20%	91.17%	91.47%
	SV	343	418	413	418	418	2075
Vid09 Frames: 423 Samples: 3709	SR	90.35%	90.05%	90.48%	90.08%	87.41%	90.16%
	SV	265	275	273	275	272	1375
Vid10 Frames: 1003 Samples: 8271	SR	81.50%	81.39%	81.37%	81.42%	80.11%	81.31%
	SV	430	437	436	437	437	2185
Vid11 Frames: 429 Samples: 4054	SR	95.93%	96.00%	96.05%	96.00%	96.05%	95.86%
	SV	315	330	329	330	330	1645

Table 3.5 shows the average processing times of different online learning algorithm for each frame of the first video sequence. As expected OL-RLS and OL-EG methods require the most computational power. Compared to the segmentation algorithm online learning methods require much less computations.

Table 3.5: Average processing times (in μs) of online learning algorithms on the first video sequence.

	Perceptron	PA-II	OL-RLS	OL-LMS	OL-EG	OL-M
Times (μs)	558.7797	525.2191	695.3380	539.8352	628.1592	562.2943

Figure 3.11 displays sample results for wildfire detection experiment. The red rectangles mark the actual smoke regions and green rectangles mark the other segments. Notice the false classifications in Vid06, Vid09 and Vid011. Overall, online learning algorithms provide a satisfactory solution a very difficult problem.

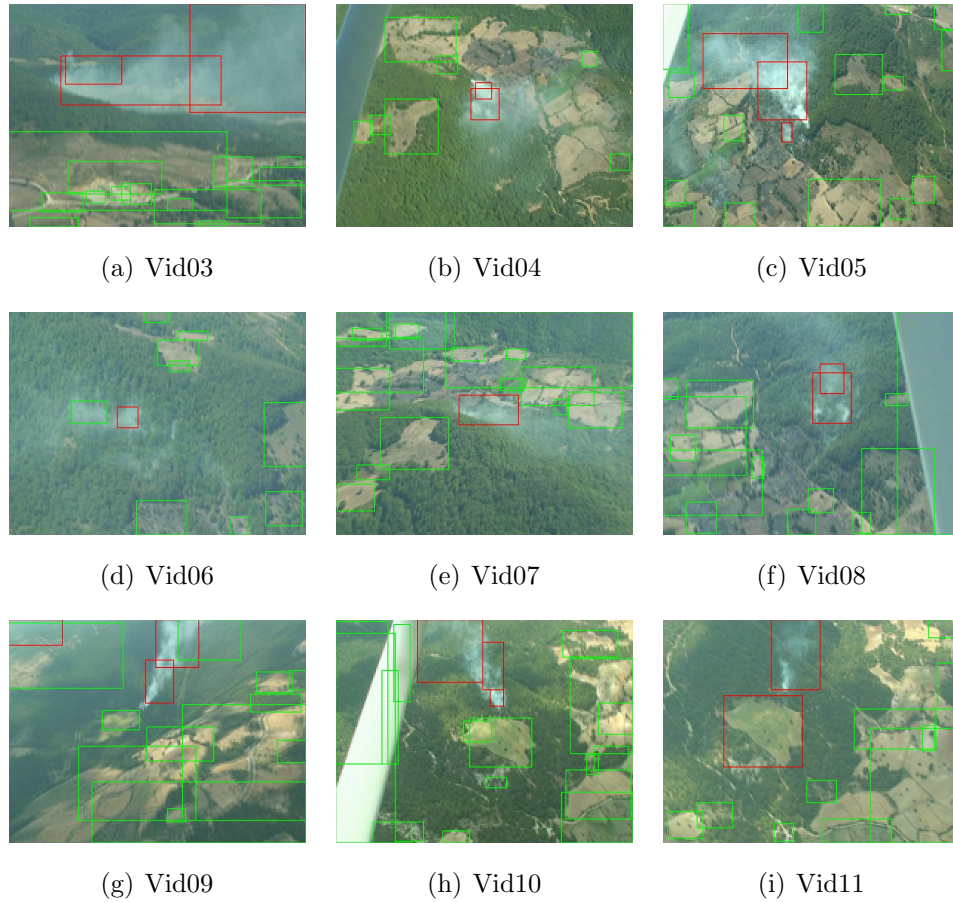


Figure 3.11: Example detection results for actual wildfire smoke images.

Chapter 4

Real-time Dynamic Texture Recognition using Random Hyperplane Projections

We propose a real-time dynamic texture recognition method using optimized local binary patterns (LBP) as the features and random hyperplane projections or deep neural networks for dimension reduction. LBP is an efficient texture representation scheme but it is computationally inefficient to use in video applications. We use random sampling to significantly reduce the computational cost of LBP method. We show that using random hyperplanes to reduce the dimension of feature vectors can decrease computational cost and preserve classification rate. We propose a real-time infrared flame detection system using the proposed feature extraction methods.

Dynamic textures are moving image regions or sequences that can be characterized by specific temporal stationarity [11]. Waves, flames, smoke, flags in the wind, flow of traffic, etc. are examples of dynamic textures. Classification of dynamic textures can be an important problem for many surveillance applications. In [74] and [20], local binary patterns (LBP) are used for recognition of dynamic textures. In these works LBPs are extended to be used in image sequences. Two new methods are proposed, named VLBP (volume LBP) and LBP-TOP (LBP-three orthogonal planes). VLBP uses three parallel planes in the time axis, and the dimensionality of the resulting feature vector is usually large. LBP-TOP

concatenates the results from three orthogonal planes in x-y and time axes to achieve smaller sized feature vectors.

In [75] view-invariant dynamic texture recognition method is proposed using bag of linear dynamics systems (LDS). They use nonlinear dimensionality reduction and clustering to use LDSs in a bag of features framework. In [76] spatio-temporal multi-resolution histograms are used for feature extraction. There are many other methods in the literature for classification, segmentation and synthesis of dynamic textures [77–80, 80–86]. Most of these method have high computational requirements and they cannot be used in real-time applications. Therefore, in this work we focus on real-time classification of dynamic textures.

Random hyperplanes are part of locally sensitive hashing based statistical dimension reduction methods [87, 88]. Deep neural networks use pre-training to achieve faster training times [89].

In recent years, many video based flame detection algorithms are developed especially for safety applications [90–93]. The algorithms developed for visible range cameras first use background subtraction to find moving regions. Then these regions are analyzed in terms of spatial irregularity and flicker frequency to determine if the movement is caused by flames. These algorithms have difficulty in dealing with false alarms caused by flame colored moving regions.

With the decreasing cost of infrared sensors, it became possible to use long and short wave infrared cameras for flame detection. Since infrared (thermal) cameras form images whose intensity depend on the temperature of the objects, they could be used to reduce most of the false alarms. In most cases the temperature of flames is higher than the surrounding environment and IR cameras can successfully detect the flickering flames. For example in Figure 4.1 the flame is clearly visible in the IR image but cannot be seen in the visible image.

One of the first methods for infrared flame detection, used near infrared images that are obtained by filtering visible range images with a 700-900 nm long-pass filter [94]. In another work hidden Markov models and wavelet analysis are used to reduce false alarms [95]. Temporal flickering of flames is modeled using Markov models. Spatial irregularity of flame regions' boundaries is modeled using the wavelet transform. In [7] a flame detection method is proposed that combines the probabilities of features obtained from visible range and long wave infrared



Figure 4.1: LWIR and visible range images of long range flame; a)LWIR image, and b) visible range image.

(LWIR) thermal images. Another multi-modal flame detection method is developed in [96]. In this method wavelet transform and nonlinear difference of Gaussians based background subtraction techniques are developed for flame detection.

The rest of the chapter is organized as follows: In Section 4.1 we review random hyperplanes, deep neural networks and local binary patterns, in Section 4.2 we present the proposed feature extraction method, Section 4.3 details the infrared flame detection algorithm, and we summarize the experimental results in Section 4.4.

4.1 Review

4.1.1 Random Hyperplanes and Deep Neural Networks

In data mining applications hashing function are used to reduce the dimensionality of the data and provide fast access to large databases. One of the dimension reduction methods used for hashing is the random hyperplane method used with inner product similarity [97]. Let us assume that $\mathbf{x}$ is an $Q \times 1$ vector representing the dynamic texture of video signal. Φ , represents a random hyperplane with the same dimensions as $\mathbf{x}$ and its elements are obtained from a standard normal distribution. The hash function for the feature vector is obtained as

$h_{\Phi} = \text{sign}(\Phi^T \mathbf{x})$. When P different random hyperplanes are used, P -bit low dimensional representation for the feature is obtained. In our application we do not use the sign function and directly use the result of the inner product to find the reduced feature vectors.

Autoencoder are one type of deep neural networks that can be used for dimensionality reduction purposes. Most important problem with neural networks is the determination of the initial weights of the layers. Deep neural networks use pre-training procedure to find the initial weights [89].

4.1.2 Local Binary Patterns

Local binary patterns (LBP) are first introduced for texture classification [19]. In this method each pixel in the image is compared to its predefined number of neighbours. If the center pixels value is larger than neighbour it is marked with 1 and marked with 0 otherwise. The binary values form the k -digit number where k is the number of neighbours. The histogram of all binary numbers corresponding to pixels form the LBP feature vector. The extension of the LBP method to image sequences is proposed in [20]. VLBP method uses three parallel planes in the time axis to form the feature vector. The binary number is obtained from the neighbours in all three planes. Therefore the dimension of the feature vector increases exponentially with the number of neighbours. LBP-TOP method decreases the dimension of the feature vector by considering three orthogonal planes around the center pixels and calculating the binary number separately for each plane. the final feature vector is obtained by concatenating the histograms corresponding to each orthogonal plane.

4.2 Feature Extraction and Classification

In our application we divide image sequence of size $W \times H \times F$ into spatially non-overlapping sub-blocks of size $W_B \times H_B \times F_B$. The blocks can overlap in time dimension. We generate a random grid of coordinates with G elements that fall within the volume corresponding to the sub-block. We calculate 3×2^k dimensional LBP-TOP feature vectors for each random coordinate for user selected k neighbours. We use standard normal distributed P random hyperplanes of size

$1 \times (3 \times 2^k)$ for dimensionality reduction. When we multiply the LBP feature vector with the random hyperplanes we get a feature vector of size $P \leq 3 \times 2^k$. For deep neural network based dimensionality reduction we use a five-layer network with sizes $(3 \times 2^k) - (1000) - (500) - (250) - (P)$. The overview of the feature extraction process is shown in Figure 4.2.

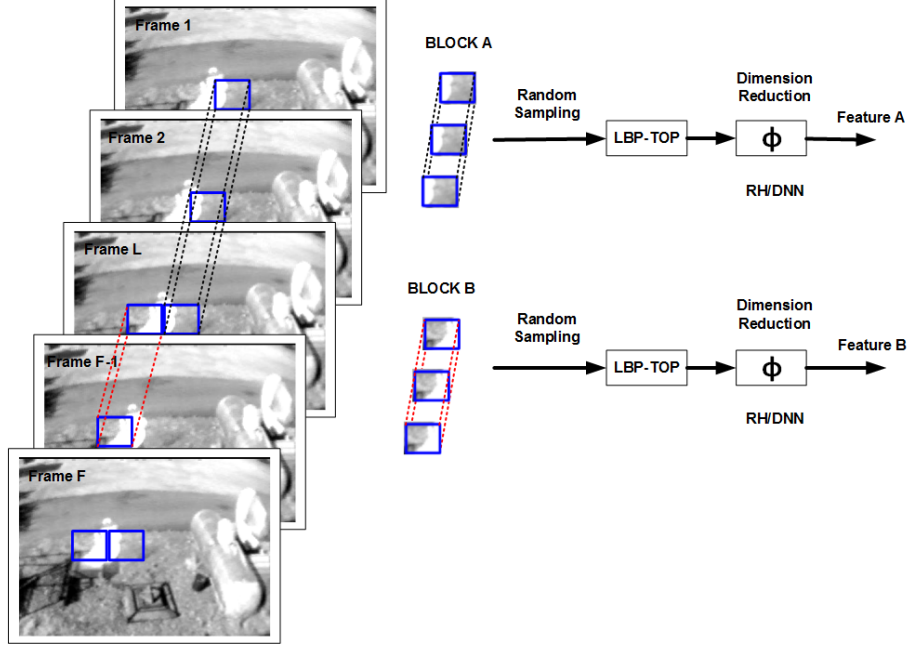


Figure 4.2: Overview of the proposed feature extraction method.

The random sampling and dimension reduction steps can provide multiple orders of magnitude speed-up in feature extraction and classification without decreasing the success rate. As an example for blocks of size $88 \times 72 \times 128$, the standard LBP-TOP requires 811008 LBP calculations. If we use only 4096 random samples we can get $198\times$ theoretical speed-up. We perform extensive experiments to validate the proposed methods. We use support vector machines (SVM) for classification [98].

4.3 Infrared Flame Detection Algorithm

We use approximate median method for moving regions detection [99]. Thresholding is applied in gray-scale images to find bright regions. After moving bright regions are detected, for each $16 \times 16 \times 64$ block the ratio of the pixels in these

regions is found. If this ratio is larger than a threshold features are extracted from this block. Each block has 16384 pixels and we use 1000 random coordinates to extract the LBP-TOP features.

In the real-time operation mode the classification starts after a learning period of 64 frames. When 64 frames are received the feature vector obtained using sampled LBP-TOP and dimension reduction is available. We use spatio-temporal blocks that overlap for 63 frames in time domain. Therefore after the initial learning period each spatio-temporal block becomes available in one frame intervals. If the block is marked as a flame-colored moving region, the resulting feature vectors are fed to the classifier.

4.4 Experimental Results

4.4.1 Dynamic Texture Recognition Experiments

We use the DynTex database in dynamic texture recognition experiments [1]. There are 3 subsets of the database used in benchmark recognition experiments. *Alpha* dataset shown in Figure 4.3 consists of 60 dynamic textures divided into 3 classes. *Beta* dataset shown in Figure 4.4 consists of 162 dynamic textures divided into 10 classes. *Gamma* dataset shown in Figure 4.5 consists of 275 dynamic textures divided into 10 classes. In the experimental setup we divide each video into $88 \times 72 \times 128$ non-overlapping spatio-temporal blocks. We limit the number of processed frames from each video to 1024. With this setup we can obtain up to 128 feature vectors from each video sequence. We define the methods compared in the experiments below:

- **LBP₄**: Full LBP-TOP method with 4-neighbourhood.
- **LBP₈**: Full LBP-TOP method with 8-neighbourhood.
- **RS₄₀₉₆LBP₄**: 4-neighbourhood LBP-TOP method calculated with randomly sampled 4096 coordinates.
- **RS₄₀₉₆LBP₈**: 8-neighbourhood LBP-TOP method calculated with randomly sampled 4096 coordinates.

- **RH₁₆LBP₄**: 4-neighbourhood full LBP-TOP method after dimension reduction using 16 random hyperplanes with a Gaussian vector of size 1×48 .
- **RH₁₂₈LBP₈**: 8-neighbourhood full LBP-TOP method after dimension reduction using 128 random hyperplanes with a Gaussian vector of size 1×768 .
- **RH₁₆RS₄₀₉₆LBP₄**: 4-neighbourhood LBP-TOP method calculated with randomly sampled 4096 coordinates after dimension reduction using 16 random hyperplanes with a Gaussian vector of size 1×48 .
- **RH₁₂₈RS₄₀₉₆LBP₈**: 4-neighbourhood LBP-TOP method calculated with randomly sampled 4096 coordinates after dimension reduction using 128 random hyperplanes with a Gaussian vector of size 1×768 .
- **DNN₁₆RS₄₀₉₆LBP₄**: 4-neighbourhood LBP-TOP method calculated with randomly sampled 4096 coordinates after dimension reduction using deep neural networks.
- **DNN₁₂₈RS₄₀₉₆LBP₈**: 4-neighbourhood LBP-TOP method calculated with randomly sampled 4096 coordinates after dimension reduction using deep neural networks.



(a) Sea



(b) Grass



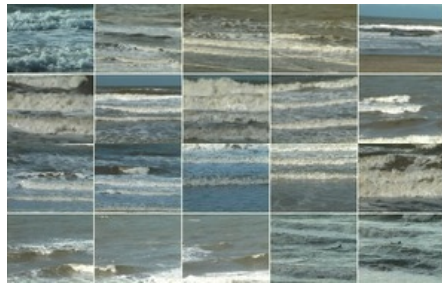
(c) Trees

Figure 4.3: *Alpha* dataset from DynTex database [1].

In Table 4.1 we show dynamic texture recognition success rates for *Alpha* dataset. In the ratio NS/A, NS is the number of total samples and A is the ratio of samples used for training. CV column displays the 10-fold cross-validation accuracy. c_i is the number of samples for the i_{th} class. From the results we see that as expected LBP_8 method shows the best performance. However, we observe that only using a 1/198 of the samples does not severely effect performance of the algorithms. $RS_{4096}LBP_4$ and $RS_{4096}LBP_8$ methods which are computationally much efficient than full methods, still have comparable recognition results. In the table bold results show the best success rates among the methods that use randomly sampled pixels. Another interesting result is that except for one case the random hyperplane method $RH_{128}RS_{4096}LBP_8$ always outperforms $RS_{4096}LBP_8$ method. The same results are confirmed on *Beta* and *Gamma* datasets in Tables 4.2 and 4.3. We can therefore conclude that proposed $RH_{128}RS_{4096}LBP_8$ is a good candidate for real-time dynamic texture recognition. Random hyperplane based methods usually outperform DNN based methods in this datasets.

Table 4.1: Dynamic texture recognition success rates (percentage) on *Alpha* dataset. In the ratio NS/A, NS is the number of total samples and A is the ratio of samples used for training. CV column displays the 10-fold cross-validation accuracy. c_i is the number of samples for the i_{th} class.

Dataset: <i>Alpha</i> , # of Samples: c_1 : 1760 c_2 : 1264 c_3 : 1600						
Method	NS/32	NS/16	NS/8	NS/4	NS/2	CV
LBP_4	95.01	97.11	98.60	98.85	99.70	99.78
LBP_8	97.03	98.62	99.49	99.50	99.84	100.00
$RH_{16}LBP_4$	92.14	96.01	98.14	98.43	99.34	99.61
$RH_{128}LBP_8$	96.42	98.36	99.13	99.22	99.72	99.91
$DNN_{16}LBP_4$	90.96	96.60	97.08	98.59	98.92	99.20
$DNN_{128}LBP_8$	96.30	97.75	98.74	98.89	99.46	99.74
$RS_{4096}LBP_4$	94.16	95.62	97.39	98.62	99.09	99.35
$RS_{4096}LBP_8$	95.14	96.88	98.55	98.90	99.49	99.74
$RH_{16}RS_{4096}LBP_4$	90.76	93.96	96.47	96.72	97.88	98.36
$RH_{128}RS_{4096}LBP_8$	96.06	97.67	98.45	98.94	99.54	99.81
$DNN_{16}RS_{4096}LBP_4$	90.80	94.45	95.86	95.04	98.11	98.33
$DNN_{128}RS_{4096}LBP_8$	95.35	97.55	96.85	98.47	98.86	99.48



(a) Sea



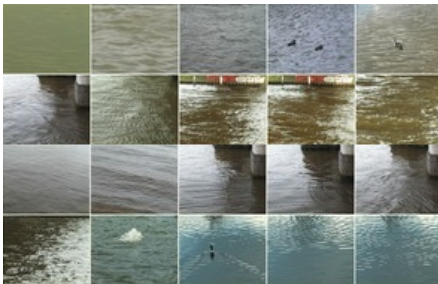
(b) Vegetation



(c) Trees



(d) Flags



(e) Calm water



(f) Fountains



(g) Smoke



(h) Escalator



(i) Traffic



(j) Rotation

Figure 4.4: *Beta* dataset from DynTex database [1].

Table 4.2: Dynamic texture recognition success rates (percentage) on *Beta* dataset. In the ratio NS/A, NS is the number of total samples and A is the ratio of samples used for training. CV column displays the 10-fold cross-validation accuracy. c_i is the number of samples for the i_{th} class.

Dataset: <i>Beta</i> , # of Samples: $c_1 : 1456$ $c_2 : 976$ $c_3 : 800$, $c_4 : 1456$ $c_5 : 1088$ $c_6 : 880$ $c_7 : 624$ $c_8 : 112$ $c_9 : 544$, $c_{10} : 816$						
Method	NS/32	NS/16	NS/8	NS/4	NS/2	CV
LBP_4	70.36	77.49	85.77	91.13	93.73	97.82
LBP_8	79.71	87.10	94.14	96.77	98.63	99.49
$RH_{16}LBP_4$	62.20	71.08	80.63	83.43	87.49	92.56
$RH_{128}LBP_8$	74.92	83.45	91.82	95.12	97.41	99.17
$DNN_{16}LBP_4$	62.40	68.86	76.56	66.68	86.67	92.68
$DNN_{128}LBP_8$	75.48	77.16	89.78	92.95	95.39	98.00
$RS_{4096}LBP_4$	68.68	73.56	81.78	86.61	90.11	94.84
$RS_{4096}LBP_8$	72.42	81.19	88.22	91.43	93.84	96.47
$RH_{16}RS_{4096}LBP_4$	58.82	66.56	76.73	79.47	82.35	88.21
$RH_{128}RS_{4096}LBP_8$	75.62	82.55	90.04	93.42	95.59	98.19
$DNN_{16}RS_{4096}LBP_4$	61.10	63.55	74.22	79.12	83.07	89.08
$DNN_{128}RS_{4096}LBP_8$	73.91	76.56	82.43	88.78	94.61	96.70



(a) Flowers



(b) Sea



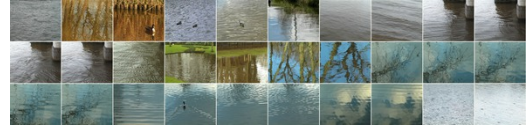
(c) Naked trees



(d) Foliage



(e) Escalator



(f) Calm water



(g) Flags



(h) Grass



(i) Traffic



(j) Fountains

Figure 4.5: *Gamma* dataset from DynTex database [1].

Table 4.3: Dynamic texture recognition success rates (percentage) on *Gamma* dataset. In the ratio NS/A, NS is number of total samples and A is the ratio of samples used for training. CV column displays the 10-fold cross-validation accuracy. c_i is the number of samples for the i_{th} class.

Dataset: <i>Gamma</i> , # of Samples: $c_1 : 1648$ $c_2 : 2816$ $c_3 : 1936$, $c_4 : 2128$ $c_5 : 112$ $c_6 : 1888$, $c_7 : 2176$ $c_8 : 1328$ $c_9 : 544$, $c_{10} : 1856$						
Method	NS/32	NS/16	NS/8	NS/4	NS/2	CV
LBP_4	72.35	76.78	82.87	86.48	91.44	94.04
LBP_8	80.76	86.32	92.08	95.49	98.26	99.47
$RH_{16}LBP_4$	65.18	69.81	73.82	77.22	79.75	83.73
$RH_{128}LBP_8$	76.89	82.79	89.82	93.41	96.27	97.91
$DNN_{16}LBP_4$	62.84	68.47	65.95	75.77	78.40	83.19
$DNN_{128}LBP_8$	71.89	79.38	80.49	89.56	92.14	95.03
$RS_{4096}LBP_4$	64.32	70.28	76.77	81.29	86.22	89.03
$RS_{4096}LBP_8$	72.18	77.18	82.51	86.70	89.69	93.03
$RH_{16}RS_{4096}LBP_4$	59.46	64.65	68.61	71.84	73.52	78.75
$RH_{128}RS_{4096}LBP_8$	74.72	79.39	85.69	89.74	92.93	94.75
$DNN_{16}RS_{4096}LBP_4$	58.61	62.71	68.39	69.07	77.00	81.43
$DNN_{128}RS_{4096}LBP_8$	71.63	74.79	82.67	86.38	88.29	90.63

In Figure 4.6 we investigate the effect of the number of random samples on the success rate for $RS_N LBP_8$ method on the *Beta* dataset. We see that after 20k samples $RS_N LBP_8$ comes very close LBP_8 's success rate which uses 811k samples. In Figure 4.7 we investigate dimensions of the random hyperplanes for $RH_M RS_{4096} LBP_8$ method on the *Beta* dataset. We see that as the dimension of the random hyperplane increases the success rate of $RH_M RS_{4096} LBP_8$ method exceeds the full dimensional $RS_{4096} LBP_8$ and comes close to the LBP_8 . When $RH_M RS_{16384} LBP_8$ method is used the success rate of random hyperplane method is also close to the success rate of the algorithm without dimensionality reduction. From this result we can say that instead of increasing the number of random samples from 4096 to 16384, we can use random hyperplanes to increase the success rate of the method. This phenomenon is not restricted to an isolated case, but holds true for all three benchmark datasets. One possible explanation is that the random hyperplane operation maps the feature vectors into a lower dimensional

space where they are better separated by the rbf kernel used in the SVM classifier. However, we do not see the same pattern when we use deep neural networks, as shown in Figure 4.8.

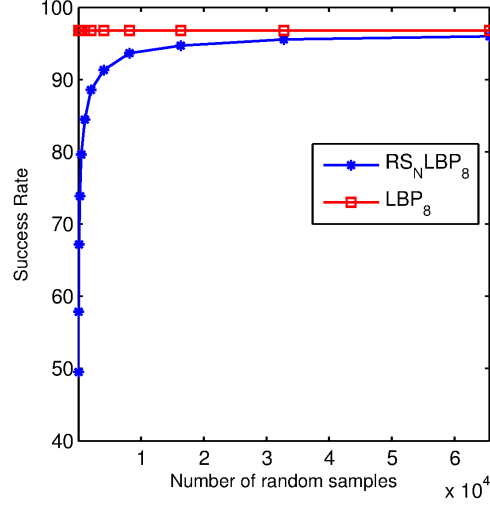


Figure 4.6: Effect of number of random samples on success rate.

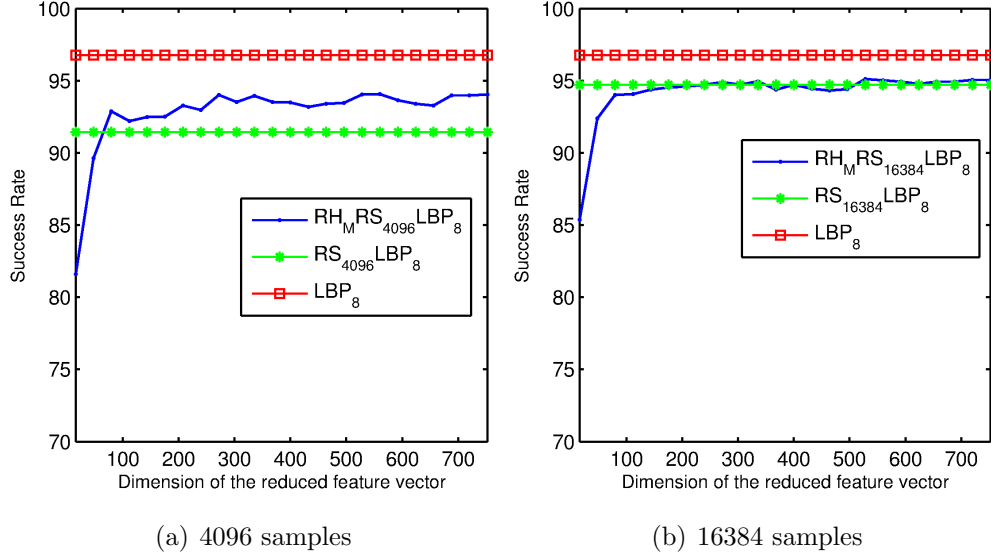


Figure 4.7: Effect of dimension of the random hyperplanes on success rate.

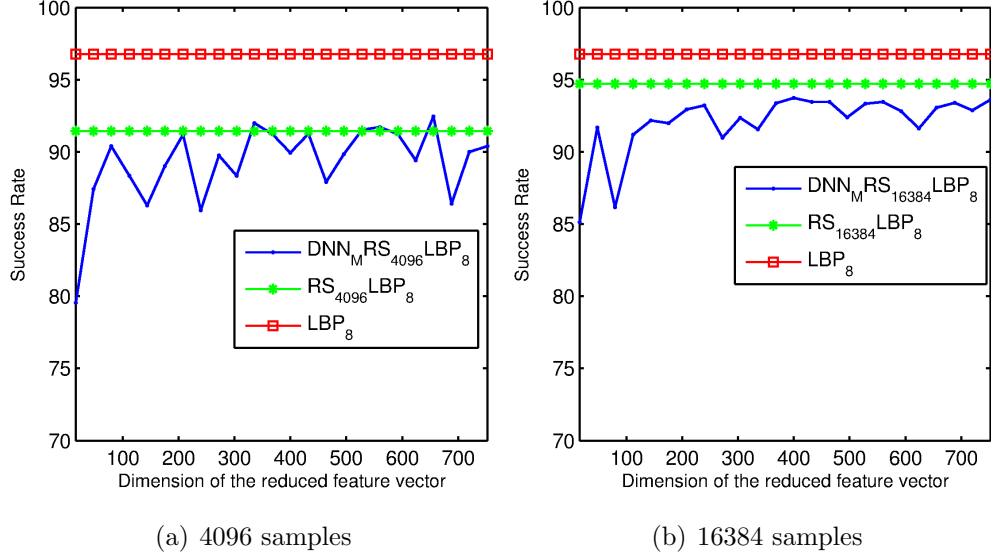


Figure 4.8: Effect of dimension of the last DNN layer on success rate.

4.4.2 Infrared Flame Detection Experiments

For infrared flame detection we use the setup described in Section 4.3. We use training and test videos that are obtained with various imaging devices and under different conditions. In the dataset we have five videos that contain actual flames (positive videos) and ten videos that contain flame colored moving regions (negative videos) that cause false alarms. We use one-fifth of the video sequences for training and the rest for testing.

Table 4.4 shows the classification results for the infrared flame dataset. In the training set we have 6663 and 5807 samples from positive and negative videos respectively. In the test set, we have 16650 positive and 10867 negative samples. In the table we show the results when we only use a fraction of the training set to see the generalization performance of the classifier. For each $16 \times 16 \times 64$ spatio-temporal block we only use randomly selected 1000 points for LBP calculations. From the results, we observe that 128-dimensional feature vectors of $RH_{128} RS_{1000} LBP_8$ perform very close to 768-dimensional feature vectors of $RS_{1000} LBP_8$. When half of the training set is used, the success rate in the test set is %94.68 for RH and %95.74 for DNN.

Table 4.4: Dynamic texture recognition success rates (percentage) on infrared flame dataset. In the ratio NS/A, NS is the number of total training samples and A is the ratio of the training set that is used. CV column displays the 10-fold cross-validation accuracy on the whole dataset. c_i is the number of samples for the i_{th} class.

Flame Dataset, Training: $c_1 : 6663$ $c_2 : 5807$ Test: $c_1 : 16650$ $c_2 : 10867$						
Method	NS/16	NS/8	NS/4	NS/2	NS/1	CV
$RS_{1000}LBP_4$	87.84	90.83	90.18	91.54	92.94	99.91
$RS_{1000}LBP_8$	95.69	95.62	95.98	96.07	96.26	99.85
$RH_{24}RS_{1000}LBP_4$	91.53	92.52	92.86	92.79	93.72	99.72
$RH_{128}RS_{1000}LBP_8$	94.59	95.25	94.34	94.68	95.17	99.98
$DNN_{24}RS_{1000}LBP_4$	93.24	93.56	92.65	94.38	95.13	99.21
$DNN_{128}RS_{1000}LBP_8$	94.53	95.06	95.17	95.74	96.56	99.90

Another important criteria for SVM classification is the number of support vectors of the model. Table 4.5 shows the number of supports vectors and total classification time (seconds) for the dataset configuration in Table 4.4. $RH_{128}RS_{1000}LBP_8$ method usually has lower number of support vectors and its classification time is significantly less than $RS_{1000}LBP_8$ method. In real-time operation mode, the proposed flame detection method can run at 10 fps on a PC with Intel I5 processor and 6 GB RAM.

Table 4.5: *Number of support vectors | total classification time (sec)* for flame detection dataset.

Flame Dataset, Training: $c_1 : 6663$ $c_2 : 5807$ Test: $c_1 : 16650$ $c_2 : 10867$					
Method	NS/16	NS/8	NS/4	NS/2	NS/1
$RS_{1000}LBP_4$	44 0.15	51 0.16	64 0.19	77 0.23	104 0.30
$RS_{1000}LBP_8$	132 6.18	154 7.16	203 9.96	244 12.15	317 14.33
$RH_{24}RS_{1000}LBP_4$	51 0.11	67 0.14	72 0.15	87 0.17	135 0.26
$RH_{128}RS_{1000}LBP_8$	50 0.36	59 0.40	64 0.44	71 0.48	104 0.68
$DNN_{24}RS_{256}LBP_4$	46 0.19	65 0.13	77 0.17	90 0.17	254 0.46
$DNN_{128}RS_{256}LBP_8$	48 0.35	64 0.45	73 0.47	88 0.55	111 0.72

In the final experiment, we compare the proposed method with the latest IR

flame detection system developed by Steven et al. [7], which uses a feature-based method. They use IIR filter based background subtraction to find moving regions, then they use Otsu thresholding to segment bright flame regions. They propose three criteria to quantify the flame existence probability. These are bounding box disorder, principal orientation disorder and histogram roughness. We implemented a tracker to measure the parameters of each candidate flame object. In Table 4.6 the proposed method is compared with with [7] in terms of correct detection rate. We see that our method has better detection rate in all the video sequences. For surveillance applications it is not necessary to detect flames in all frames, it is important not to miss any flame detection in a specified time. The most important problem for surveillance applications is the reduction of false alarms. Table 4.7 shows the false detection rates of the methods on videos that contain false alarm sources. We see that the proposed method significantly reduces false positive detections compared to [7]. There are still some nuisance alarms issued to some video sequences but they can be removed with additional temporal processing. This can include assigning levels to each alarm according to the consecutive number of frames that it is issued. But this method might not work for the system described in [7], since the false alarm rates are too high for some sequences.

Table 4.6: Proposed method is compared with [7] in terms of correct detection rates.

Video	Correct Classification Rates	
	Our Method	Steven et al [7]
Video01	697 / 935 (% 74.55)	431 / 935 (% 46.10)
Video02	935 / 935 (% 100.00)	721 / 935 (% 77.11)
Video03	935 / 935 (% 100.00)	905 / 935 (% 96.79)
Video04	335 / 335 (% 100.00)	335 / 335 (% 100.00)
Video05	156 / 157 (% 99.36)	152 / 157 (% 96.82)

Table 4.7: Proposed method is compared with [7] in terms of false detection rates

	False Alarm Rates	
Video	Our Method	Steven et al [7]
Video06	048 / 935 (% 5.13)	040 / 935 (% 4.28)
Video07	023 / 507 (% 4.54)	016 / 507 (% 3.16)
Video08	006 / 512 (% 1.17)	030 / 512 (% 5.86)
Video09	005 / 935 (% 0.53)	616 / 935 (% 65.88)
Video10	053 / 801 (% 6.62)	188 / 801 (% 23.47)
Video11	000 / 935 (% 0.00)	000 / 935 (% 0.00)
Video12	023 / 550 (% 4.18)	195 / 550 (% 35.45)
Video13	013 / 737 (% 1.76)	249 / 737 (% 33.79)
Video14	000 / 150 (% 0.00)	070 / 150 (% 46.67)
Video15	000 / 935 (% 0.00)	208 / 935 (% 22.25)



(a) True alarms



(b) False alarms

Figure 4.9: Sample snapshots from flame detection dataset.

Figure 4.9 shows samples of correctly classified and misclassified examples from the real-time recognition dataset. In the experiments, our videos are recorded with Xenics Bobcat-320 short wave IR camera that works in 0.9 - 1.7 μ m wavelength range and Xenics Gobi-384 long wave IR camera that works in 8 - 14 μ m wavelength range. The other videos are obtained from open sources and FIRESENSE project database [3, 7, 41, 72].

Chapter 5

Wildfire Detection with PTZ Cameras using Panoramic Backgrounds

5.1 Introduction

We consider continuously moving cameras for early detection of wildfires. Current wide-area wildfire detection camera systems scan predefined preset positions and run the detection algorithm suited to stationary cameras. This can increase the detection time substantially when full 360 coverage of the area is needed. Therefore, we propose a wildfire detection system using continuously moving cameras. In [100] a real-time method for panoramic image registration is proposed. When forming the panorama they also generate a set of key frames and during the real-time registration process, they register each new image to the closest key frame. In this way, they overcome the error accumulation problem that can result from successive registrations to the panorama.

In [101], they use scale invariant feature transform (SIFT) and dynamic programming. SIFT is used for feature extraction and dynamic programming is used to stitch the panorama without artifacts. There are many other methods for panoramic image generation and its applications [102–109], but the key-frame based method in has the highest potential for the wildfire detection problem.

We use a hybrid registration method using speeded-up robust features (SURF)

and mutual information to form the panoramic background and select the key-frames. In the normal mode, we use SURF for matching each image to the previous frame, when the number of inliers between the images is not enough we use mutual information based optimization to find the transformation matrix. In the real-time operating mode, we again use a mutual information based similarity measure to match the current image to one of the key frames. We then update the key frames that correspond to the background using the matched section of the current image.

After finding the moving regions we track them and form a spatio-temporal video block that falls into the same region of the key frame. We then extract features from this block using local binary patterns (LBP). We feed the features to a support vector machine to decide if the block belongs to a wildfire region.

5.2 Panorama Generation

5.2.1 Robust Features

We use SURF for feature extraction and feature correspondence. RANSAC is used for finding the homography between a pair of images. After finding the homography matrix we warp one of the images using a projective transform and add the next image to overlap the matching features. SIFT features are extracted around interest points and they can be used to match descriptors across different views of an image [21]. They are invariant to orientation and scale and they are robust to most affine viewpoint changes, different illuminations and noise. The first step in SIFT feature extraction is the generation of Gaussian and Difference of Gaussian (DOG) pyramids. The candidate keypoints are determined by finding the local maximum/minimum of the DOG images. After removing unstable keypoints an orientation and a scale are assigned to each of the remaining keypoints. The orientations of the pixels around the keypoint are used to create an orientation histogram. SIFT method is computationally costly, and therefore, the SURF method is generally preferred in real-time applications. SURF uses integral images for convolutions and Haar wavelets for features [110].

5.2.2 Mutual Information

We optimize a mutual information based cost function to correctly register images when SURF method fails to find any inliers. For two random variables A and B the mutual information is defined as follows:

$$MI(A, B) = H(A) + H(B) - H(A, B), \quad (5.1)$$

where $H(A)$ is the marginal entropy of A , $H(B)$ is the marginal entropy of B and $H(A, B)$ is the joint entropy. In image registration problems entropy functions are estimated from image histograms and joint histograms which can be calculated using kernel density estimation methods [22]. Shannon entropy of a discrete random variable can be defined as follows:

$$H(A) = - \sum_x p_A(x) \log(p_A(x)). \quad (5.2)$$

For image registration we first need to estimate the probabilistic distribution of the pixel values of images:

$$p_A(z) = \frac{1}{N} \sum_x R(z - A(x)), \quad (5.3)$$

where N is the number of samples that are used for the density estimation. In the usual histogram estimation R is the Kronecker delta function, and in kernel density applications R is a kernel such as, Gaussian or B-spline functions. The joint density for two random variables can be also estimated as follows:

$$p_{A,B}(z, r) = \frac{1}{N} \sum_x R(z - A(x)) R(r - B(x)). \quad (5.4)$$

Using the density estimates, mutual information for images can be approximated as follows [22]:

$$\begin{aligned} MI(I_V, I_R) \approx & \frac{-1}{N_B} \left\{ \sum_{I_V(i) \in B} \ln \frac{1}{N_A} \sum_{I_V(j) \in A} R(I_V(i) - I_V(j)) \right\} + \\ & \frac{-1}{N_B} \left\{ \sum_{I_R(i) \in B} \ln \frac{1}{N_A} \sum_{I_R(j) \in A} R(I_R(i) - I_R(j)) \right\} - \\ & \frac{-1}{N_B} \left\{ \sum_{I_W(i) \in B} \ln \frac{1}{N_A} \sum_{I_W(j) \in A} R(I_W(i) - I_W(j)) \right\}, \end{aligned} \quad (5.5)$$

where R is the Gaussian window used in kernel density estimation, $I_W(i) = [I_V(i) \ I_R(i)]$. N_A and N_B denote the sizes of different sample sets. For optimization of the mutual information cost function various methods can be used, gradient descent, quasi-Newton or genetic algorithms are standard choices.

5.2.3 Finding Key-frames

We use a projective transformation matrix that has eight unknowns:

$$\mathbf{T} = \begin{bmatrix} T_{11} & T_{12} & T_{13} \\ T_{21} & T_{22} & T_{23} \\ T_{31} & T_{32} & 1 \end{bmatrix}. \quad (5.6)$$

The parameters of the matrix can be determined using random samples consensus (RANSAC) [111]. If we define the transformation matrix between two successive images as, $\mathbf{T}_{n,n-1}$, the total transformation between the first and the n^{th} frame can be defined as follows:

$$\mathbf{T}_{n,1} = \mathbf{T}_{n,n-1} \mathbf{T}_{n-1,n-2}, \dots, \mathbf{T}_{2,1}. \quad (5.7)$$

Using these transformations each image can be mapped to a reference coordinate frame. To find the key-frames we look at the ratio of the overlapping pixels between successive frames if it is smaller than a threshold we add the current frame to the list of key-frames. In Figure 5.1 key-frames and the resulting panorama is shown for a sample video sequence. The original sequence has 260 frames, but in the panorama it can be represented with only four key-frames when %50 overlap is assumed between frames.



(a)



(b)

Figure 5.1: a) Key-frames, and b) resulting panorama from a sample video clip. Key-frames are marked with rectangles in the panorama. The video is taken from [2].

As we can see from the figure, as we near the edges of the panorama, projective transform results in large warped images. We can use cylindrical projections to prevent the panorama from becoming too large. When the camera is only panning or tilting without rotations or zooming, the camera movement can be represented as translations in cylindrical coordinates [111]. For Cartesian image coordinates (x_c, y_c) , their cylindrical projections, (x_s, y_s) are given by:

$$\begin{aligned} x_s &= f \arctan \left(\frac{x_c}{f} \right), \\ y_s &= f \frac{y_c}{\sqrt{x_c^2 + y_c^2}}, \end{aligned} \tag{5.8}$$

where f is the focal length of the camera. Figure 5.2 shows results of cylindrical projections on the same video.



(a)



(b)

Figure 5.2: a) Key-frames, and b) resulting panorama in cylindrical coordinates from a sample video clip. Key-frames are marked with rectangles in the panorama. The video is taken from [2].

SURF method works best, when the images have well-defined edges as in the above examples. However, we observed that it can fail to register images in videos recorded with wildfire surveillance cameras in less than ideal conditions as shown in Figure 5.3. As we can see the images are blurry and the edges are not very distinctive. With these images, the SURF method fails to find enough inliers (matching feature locations), but we can use mutual information (MI) based optimization to register the images.



Figure 5.3: Wildfire surveillance images: a) current frame, and b) next frame.

After finding the key-frames, the real-time processing needs to register each new image to a key-frame and update the background. For the first new frame we search the whole key-frame dataset, for the other frames we look for a subset of the key-frames that are close to the previously selected key-frame. We observed that matching the current frame to the key-frames is best accomplished using MI, instead of SURF features. We show an example of this for the image set in Figure 5.4. The panorama scene provides 180° coverage with 1300 frames which is represented by 18 key-frames with %75 overlap between them. Figure 5.5 shows the frame index versus key-frame index for this video sequence for both MI and SURF methods. We see that MI method shifts between the key-frames without any distortions, while SURF method oscillates between neighboring frames. When we use the MI method, the reference key-frame is the same for 60 frames on the average. This means that the locations of the wildfire region will be the same for this time interval. We can then extract spatio-temporal features from the subset of the video sequence that characterize the wildfire behavior.

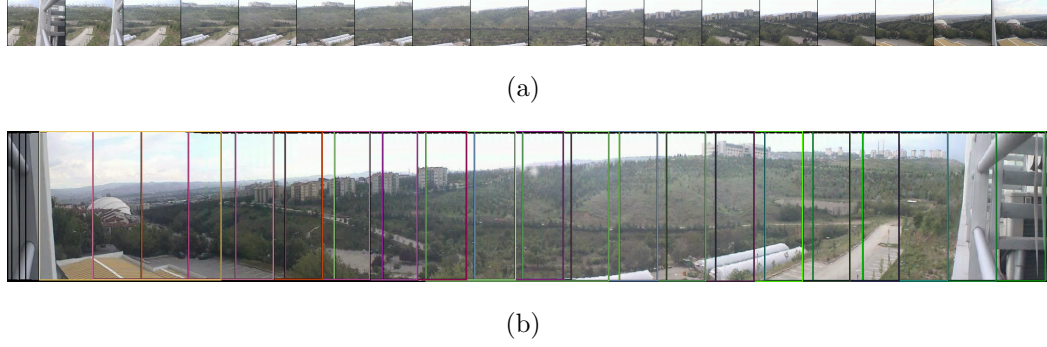


Figure 5.4: a) Key-frames, and b) resulting panorama in cylindrical coordinates from a sample video. Key-frames are marked with rectangles in the panorama.

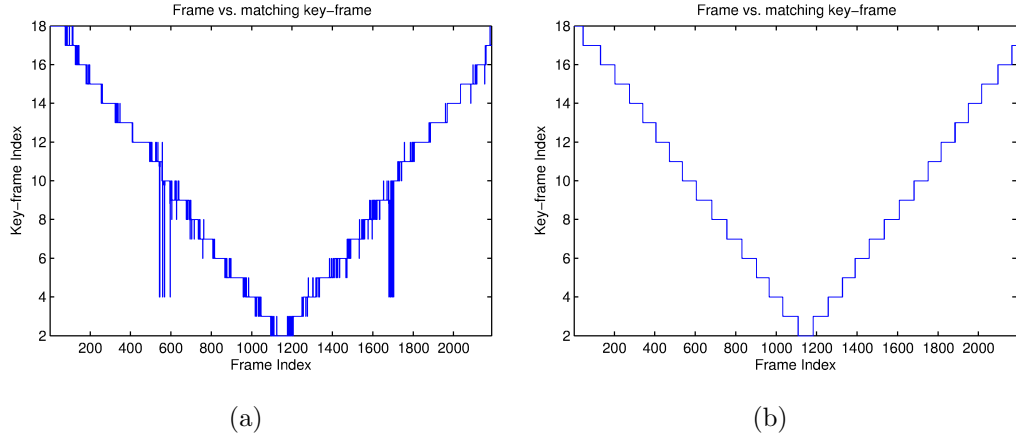


Figure 5.5: Frame index versus key-frame index for real-time operation using the background panorama of Figure 5.4: a) SURF method, b) MI method.

5.3 Wildfire Detection Algorithm

The wildfire detection algorithm consists of the following main steps; background subtraction using registered images, color analysis, robust feature analysis, spatio-temporal feature extraction and classification. We use an infinite impulse response (IIR) filter based background subtraction method defined in [112]. We use a large update parameter to adapt to slow update periods caused by the continuous movement of the camera. After background subtraction we perform color analysis of detected regions to determine candidate smoke regions. Wildfire smoke usually has a gray-to-white color that results from burning vegetation. We use thresholds

in YUV color space to determine color content of regions. The color condition is defined as follows [13]:

$$M_C = Y > T_Y \quad \& \quad |U - 128| < T_U \quad \& \quad |V - 128| < T_V, \quad (5.9)$$

where Y , U , V denote separate channels of the color space, T_Y , T_U and T_V are predetermined thresholds and M_C is the color mask for the whole image.

Another observation about the smoke regions is that they are usually smoother than the surrounding background covered with trees and irregular terrain. We exploit this observation by masking the regions around the SURF features according to their scale values.

5.3.1 Feature Extraction

For each spatio-temporal block if the ratio of the pixels satisfying the color condition is above a threshold, this block is further processed for feature extraction and classification. We use a local binary patterns (LBP) based feature extraction method also described in Chapter 4.

In our application we accumulate the detection results from a smoke region that is registered to the same key-frame. The dimension of the block is $W_B \times H_B \times F_B$. We generate a random grid of coordinates with G elements that fall within the volume corresponding to the spatio-temporal smoke region. We calculate 3×2^k dimensional LBP feature vectors for each random coordinate for user selected k neighbours. We use standard normal distributed P random hyperplanes of size $1 \times (3 \times 2^k)$ for dimensionality reduction. When we multiply the LBP feature vector with the random hyperplanes we get a feature vector of size $P \leq 3 \times 2^k$. The overview of the wildfire detection system is shown in Figure 5.6.

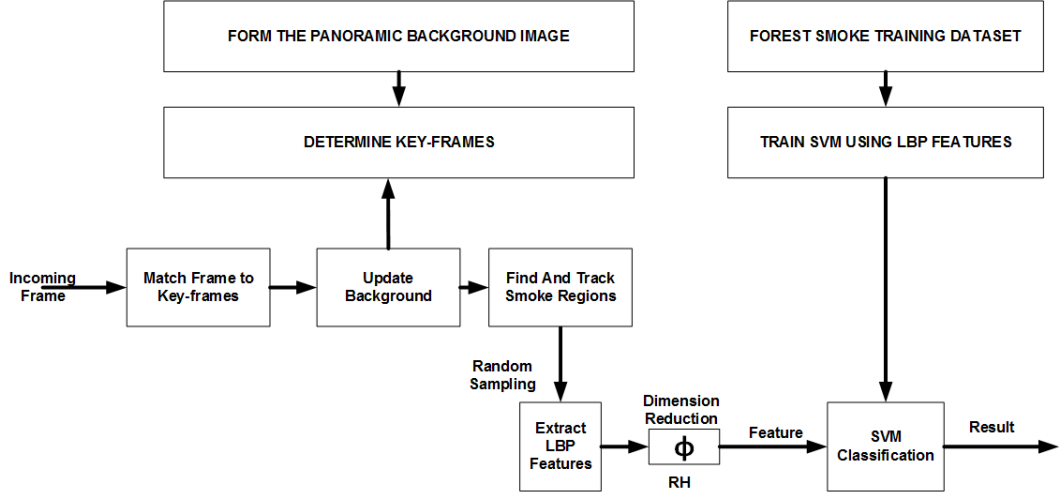


Figure 5.6: Overview of the proposed panoramic wildfire detection and feature extraction method.

The random sampling and dimension reduction steps can provide multiple orders of magnitude speed-up in feature extraction and classification without decreasing the success rate. As an example for blocks of size $16 \times 16 \times 64$, the standard LBP requires 16384 LBP calculations. If we use only 1024 random samples, we can get $16\times$ theoretical speed-up. We use support vector machines (SVM) for classification [98].

5.4 Experimental Results

In the first part of the experiments we present the classification results for the feature extraction process. We use training and test videos that are obtained from various wildfire surveillance towers and under different conditions. In the dataset we have 13 videos that contain actual smoke (positive videos) and 12 videos that contain smoke colored moving regions (negative videos) that cause false alarms. We use one-fifth of the video frames for training and the rest for testing. We define the methods compared in the experiments below:

- **LBP₄**: Full LBP method with 4-neighbourhood.
- **LBP₈**: Full LBP method with 8-neighbourhood.

- **RS₁₀₀₀LBP₄**: 4-neighbourhood LBP method calculated with randomly sampled 1000 coordinates.
- **RS₁₀₀₀LBP₈**: 8-neighbourhood LBP method calculated with randomly sampled 1000 coordinates.
- **RH₁₆RS₁₀₀₀LBP₄**: 4-neighbourhood LBP method calculated with randomly sampled 1000 coordinates after dimension reduction using 16 random hyperplanes with a Gaussian vector of size 1×48 .
- **RH₁₂₈RS₁₀₀₀LBP₈**: 4-neighbourhood LBP method calculated with randomly sampled 1000 coordinates after dimension reduction using 128 random hyperplanes with a Gaussian vector of size 1×768 .

Table 5.1 shows the classification results for the wildfire smoke dataset. In the training set we have 34798 and 57507 samples from positive and negative videos respectively. In the table we show the results when we only use a fraction of the training set to see the generalization performance of the classifier. For each $16 \times 16 \times 64$ spatio-temporal block we only use randomly selected 1000 points for LBP calculations. From the results we see that 128-dimensional feature vectors of $RH_{128}RS_{1000}LBP_8$ perform very close to 768-dimensional feature vectors of $RS_{1000}LBP_8$. When one-eight of the training set is used the success rate in the test set is %96.68 for RH, which is quite satisfactory for this problem.

Table 5.1: Wildfire smoke recognition success rates (percentage). In the ratio NS/A, NS is the total number of training samples and A is the ratio of the training set that is used. c_i is the number of samples for the i_{th} class.

Smoke Dataset: $c_1 : 34798$ $c_2 : 57507$					
Method	NS/128	NS/64	NS/32	NS/16	NS/8
$RS_{1000}LBP_4$	87.02	90.19	92.72	94.09	95.17
$RS_{1000}LBP_8$	87.50	89.67	91.13	94.15	97.01
$RH_{24}RS_{1000}LBP_4$	84.92	87.85	89.02	90.08	91.11
$RH_{128}RS_{1000}LBP_8$	88.29	91.54	94.08	95.66	96.68

Another important criteria for SVM classification is the number of support vectors of the model. Table 5.2 shows the number of supports vectors and total classification time (seconds) for the dataset configuration in Table 5.1. $RH_{128}RS_{1000}LBP_8$

method usually has lower number of support vectors and its classification time is significantly less than $RS_{1000}LBP_8$ method.

Table 5.2: *Number of support vectors | total classification time (sec)* for smoke detection dataset.

Smoke Dataset: $c_1 : 34798$ $c_2 : 57507$										
Method	NS/128		NS/64		NS/32		NS/16		NS/8	
$RS_{1000}LBP_4$	181	1.73	332	3.24	576	5.57	1049	10.11	1916	17.87
$RS_{1000}LBP_8$	321	65.09	466	77.82	687	132.28	1023	170.59	1798	230.60
$RP_{24}RS_{1000}LBP_4$	210	1.44	409	2.64	793	5.08	1507	9.52	2856	15.73
$RP_{128}RS_{1000}LBP_8$	202	4.15	327	6.62	583	11.74	975	20.73	1698	38.12

It is also possible to use wildfire detection methods proposed in Chapters 2 and 3 in the final classification process. We compare these methods using a common dataset. The results are shown in Table 5.3. We see that LBP based method and the method described in Chapter 2 method have similar performances and they have better recognition rates than the method described in Chapter 3 method.

Table 5.3: LBP based method compared to methods in Chapters 2 and 3 on a common dataset.

Smoke Dataset: Training: $c_1 : 5837$ $c_2 : 13661$ Test: $c_1 : 26406$ $c_2 : 39748$					
Method	NS/16	NS/8	NS/4	NS/2	NS
$RP_{128}RS_{1000}LBP_8$	91.64	93.48	94.50	95.21	95.73
Chapter 2 Method	94.14	94.74	95.62	95.28	95.61
Chapter 3 Method	87.27	88.09	88.93	89.00	89.58

In the next part, we demonstrate sample detection results from videos recorded with continuously panning cameras. Figure 5.7 shows detection results for the video sequence in Figure 5.4.



Figure 5.7: Wildfire detection results for two different frame and key-frame pairs using the video sequence from Figure 5.4. Top-left image is the current frame, top-right image is the corresponding key-frame, bottom-left image is the registered current image, bottom-right image is the candidate smoke region.

Figure 5.8 shows the results from a 360° coverage panorama. The panorama uses 113 key-frames to represent a full scan of the camera consisting of 9000 frames. We see that the system detects the wildfire smoke even when the image quality is poor with the help of the MI method.



Figure 5.8: Wildfire detection results for two different frame and key-frame pairs using the 360° coverage video sequence. Top-left image is the current frame, top-right image is the corresponding key-frame, bottom-left image is the registered current image, bottom-right image is the candidate smoke region.

Chapter 6

Multi-modal Image Registration using Mutual Information and Wavelet Measures

6.1 Introduction

There are various methods in the literature for registration of IR and visible images [23, 113–116]. Feature based methods try to find features that have the same characteristics in IR and visible images. Edge and gradient information are the most frequently used features. There are also line and segment based methods that try to match geometric shapes between images. Some methods use silhouettes or motion information for image registration.

Mutual information on the other hand uses the whole images to extract statistical features that can be used to compare multi-modal images. In [22], a medical volume registration method using mutual information is developed. Mutual information based registration methods use entire images to extract statistical features that can be used to compare multi-modal images. In [22], mutual information and its derivative are estimated using Parzen windows. It is shown that it is sufficient to use a small set of random number of pixels for density estimation. Gradient descent is used to optimize the mutual information cost function. The method has shown excellent performance in registration of medical images and volumes, but the cost function becomes irregular for registration of natural

images.

In [117] a mutual information based real-time image registration method is described. This method focuses on real-time tracking applications and the cost function is optimized using a Newton-like minimization algorithm which is based on a modified Hessian matrix computation method. Although this method is computationally efficient for tracking objects for which frame-to-frame displacement is small, it becomes slow for full-scale registration of multi-modal images in which high resolution mutual estimates are required.

In [118], normalized gradient fields are proposed as an alternative to mutual information based methods. In this method dot or cross products between the normalized gradients of two images are used as the distance measures. A Gauss-Newton method is used as the optimization algorithm. This simple method has demonstrated good performance in medical images but it hasn't been tested with natural images.

An intensity and edge orientation based visible and infrared image registration method is described in [23]. Joint histogram of edge orientation difference of images are calculated to define an entropy based cost function. The cost function is scaled using a weighting function that represents the total contrast between the edge orientations of images. Since the proposed cost function is not differentiable they use the downhill-simplex method to minimize the cost function. The proposed method has comparable performance to mutual information based methods but its computational cost can become very high.

In [119], wavelet energy maps of multi-modal images instead of actual intensity values are used to calculate mutual information. Wavelet energy maps are obtained from redundant wavelet transform which preserves the size of the images after transformation. But this method of applying mutual information directly to gradient or wavelet images is criticized in [120] claiming that this would create a cost function with a "narrow attraction range" and it would discard important information from intensity values. To remedy this problem in [120], a combined cost function is proposed that consists of mutual information and gradient measures. Mutual information is used to measure the statistical similarity between images. It is indicated that mutual information fails to obtain any information from the spatial similarity of images. Therefore, they use gradient information

to add spatial information into the combined cost function. The gradient measure uses magnitude and orientation of the gradients to determine the spatial similarity. In this implementation Powell’s method [121] is used to optimize the combined cost function.

We propose two new cost functions and an efficient optimization algorithm to register multi-modal images. The first cost function uses log-polar transforms in the Fourier domain together with mutual information to recover scale and orientation parameters. The second cost function combines mutual information and redundant wavelet transform based similarity measures to recover translation parameters. The deficit of mutual information in measuring spatial similarity is solved using the redundant wavelet transform. Amplitudes and orientations of different sub-bands are used to form the similarity measure. The wavelet based measure results in a smoother cost function than gradient based similarity measures. We use a genetic algorithm to minimize the non-smooth cost-function of the rotation and scale parameters. We also propose a lifted projection based optimization strategy to minimize the cost function of translation parameters.

The rest of the chapter is organized as follows; In Section 6.2 we introduce the framework for multi-modal image registration. In Section 6.3, we formulate the cost function and optimization method for recovering translations. Section 6.4, explains the cost function and the optimization method for scale and rotation. The experimental results are presented in Section 6.5.

6.2 IR and Visible Image Registration

Multi-modal image registration optimization tries to find a transformation T that maximizes the similarity between images using a suitable measure. The optimization problem can be formulated as follows:

$$\arg \max_T \mathcal{C}(I_V, T(I_R, \mathbf{p})), \quad (6.1)$$

where I_V and I_R are visible and IR images, respectively, and $\mathcal{C}$ is cost function, and $\mathbf{p} = [s, \phi, t_x, t_y]$ is the parameter set of the transformation matrix T which

can be written as follows:

$$T = \begin{bmatrix} s \cos \phi & s \sin \phi & t_x \\ -s \sin \phi & s \cos \phi & t_y \\ 0 & 0 & 1 \end{bmatrix}, \quad (6.2)$$

where ϕ is the rotation parameter, s is the scale parameter and (t_x, t_y) are the translation parameters, respectively.

We focus on the “Similarity Transform” [122], which does not distort the shape of objects. Similarity transform based methods determine only scale, rotation and translation parameters. The optimization problem is then to find the best $\mathbf{p} = [s, \phi, t_x, t_y]$ parameter set according to a suitable cost function. The flowchart of the proposed registration algorithm is given in Figure 6.1. In medical image registration the intensity difference between images of different modalities is not very large and all four registration parameters can be obtained using a simple gradient descent based optimization procedure. However, we observed that the cost function is neither unimodal nor smooth for natural images when all four parameters are used in the registration procedure. Therefore we first determine scale and rotation and then find translations in our application.

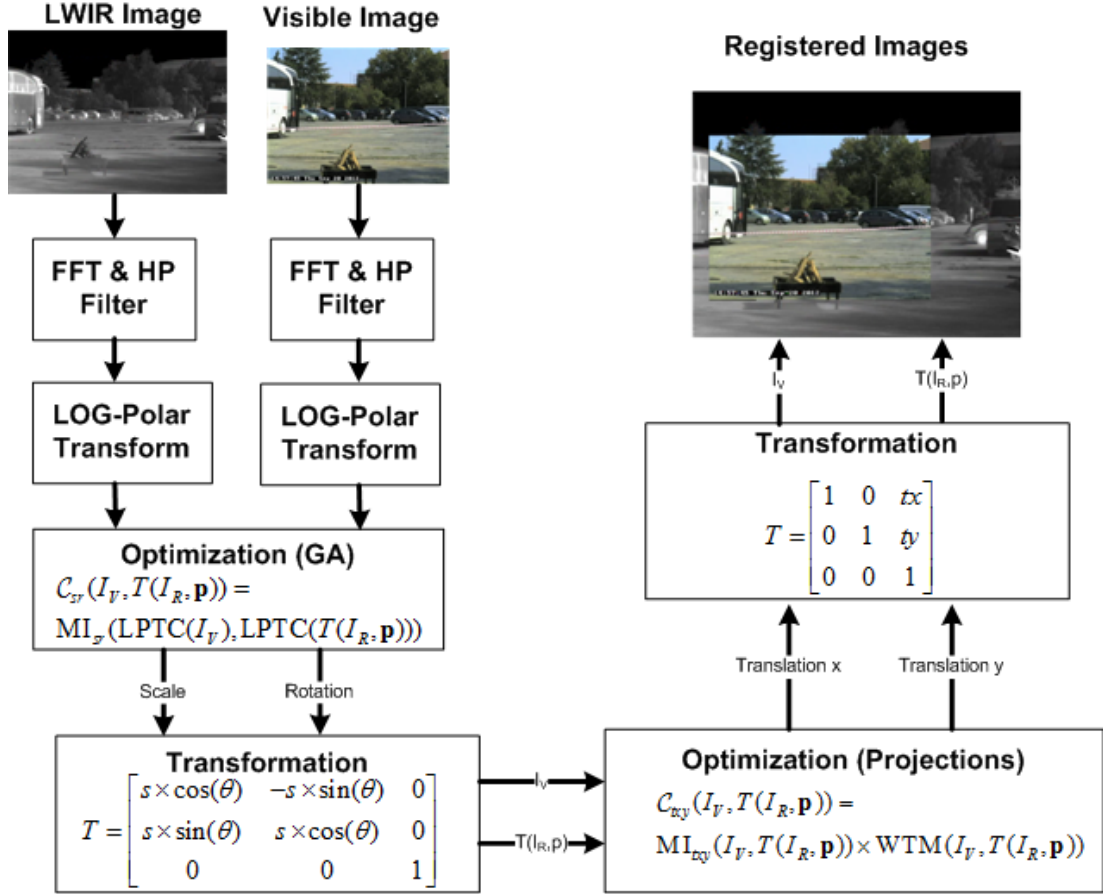


Figure 6.1: Flowchart of the proposed method.

6.3 Cost Function and Optimization for Translations

6.3.1 Cost Function

The cost function to recover translations depends on mutual information and wavelet transform similarity:

$$\mathcal{C}_{txy}(I_V, T(I_R, \mathbf{p})) = \text{MI}_{txy}(I_V, T(I_R, \mathbf{p})) \times \text{WTM}(I_V, T(I_R, \mathbf{p})), \quad (6.3)$$

where MI_{txy} and WTM are the mutual information and wavelet transform similarity measures, respectively. In the rest of the chapter we will use $\mathcal{C}(I_V, T(I_R, \mathbf{p})) = \mathcal{C}(\mathbf{p})$ as a short hand notation for cost functions.

It is also possible to write mutual information in the following form [123]:

$$MI(I_V, I_R) = H(I_V) + H(I_V|I_R). \quad (6.4)$$

In Equation 6.4, the first term does not change during the registration process. We can then re-write the cost function as follows [123]:

$$\begin{aligned} H(I_V|I_R) &= \sum_{y=1}^N p(I_R = y) \times \\ &\left\{ \sum_{x=1}^N p(I_V = x|I_R = y) \log p(I_V = x|I_R = y) \right\} \\ &= \sum_{y=1}^N p(I_R = y) f \left(\frac{p(I_V=1, I_R=y)}{p(I_R=y)}, \dots, \frac{p(I_V=N, I_R=y)}{p(I_R=y)} \right), \end{aligned} \quad (6.5)$$

where f is the entropy function. In [123] it is shown that different functions such as, geometric mean, $\mathcal{L}_p$ norms, and $\mathcal{L}_p$ deviations can be used instead of the entropy function. In this work we experiment with distance functions between the marginal distributions and the uniform distribution instead of the entropy function. The details of the selection of the distance function are given in Appendix A.

Figure 6.2 shows two visible range images and their joint histograms. One of the images is the translated version of the other. When the images are registered the joint histogram values are densest along the diagonal line. Registration is achieved by minimizing the cost function introduced in (6.3). In Figure 6.3 registration results are shown for a visible range and LWIR (long wave infrared) image pair. When the images are registered the joint histogram becomes denser along the main diagonal line compared to the unregistered case. Since the images are multi-modal the joint histogram is more scattered compared to Figure 6.2.

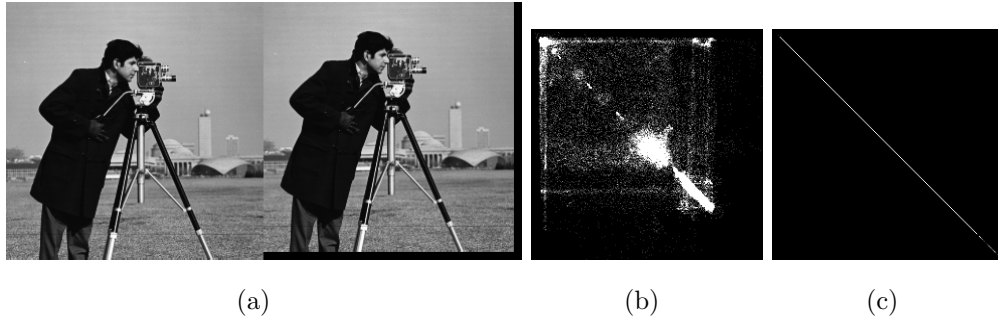


Figure 6.2: a) Original and translated visible range images, b) the joint histogram of unregistered images, and c) the joint histogram of registered images.

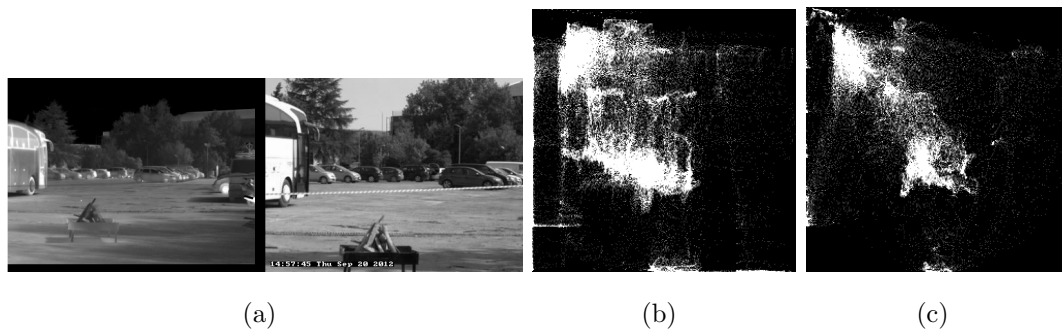


Figure 6.3: a) Original and translated visible range and LWIR images, b) the joint histogram of unregistered images, and c) the joint histogram of registered images.

Mutual information is successfully used in many multi-modal registration applications but it does not take spatial information of images into account. In [120] mutual information is multiplied by a gradient measure to account for the spatial similarity between the images.

In this application, we replaced the gradient measure with various wavelet transforms and compared redundant wavelet transform (RWT) [124], dual-tree complex wavelet transform (DT-CWT) [125], and a feature based similarity measure (FSIM) [126] to obtain a robust multi-modal spatial similarity measure. For RWT and DT-CWT measures we use the method introduced in [120] to obtain the measure. FSIM uses gradient information and phase congruency maps to measure the degree of similarity between images. Figure 6.5 displays the similarity measure values versus xy-translations for different similarity measures. DT-CWT measure has ripples that present too many local minima. FSIM measure is quite smooth but RWT has smaller ambiguous region than FSIM. The best performance for this case is obtained using the RWT measure which is smooth and captures the true scale value at the global minimum.

Figure 6.4 displays the similarity measure values vs scale for different similarity measures. DT-CWT measure has ripples that present too many local minima. FSIM measure is quite smooth but it has a global minimum at 0.6 that is smaller than the ground truth scale value that is 0.65. The best performance for this case is obtained using the RWT measure which is smooth and captures the true scale value at the global minimum. The same experiment is performed for similarity

value versus xy-translations in Figure 6.5. For this case DT-CWT still has ripples and RWT has smaller ambiguous region than FSIM measure.

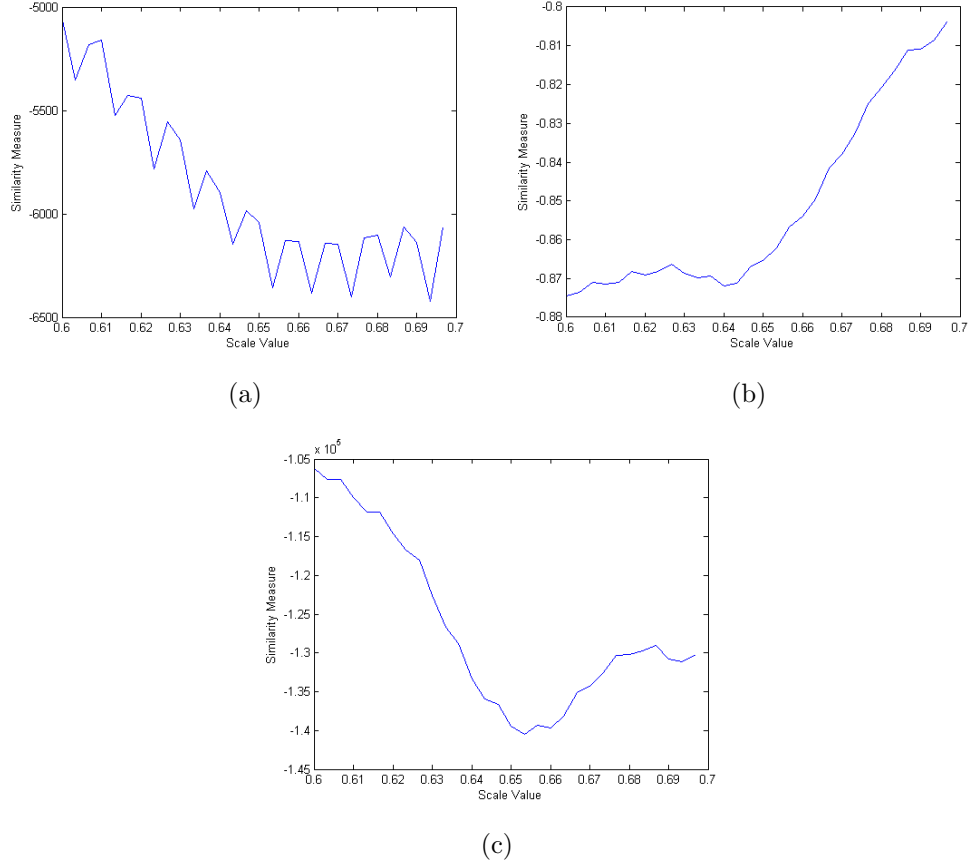


Figure 6.4: Similarity value vs scale for a) DT-CWT, b) FSIM, and c) RWT similarity measures. The images that are used to calculate the measures are given in Figure 6.3.

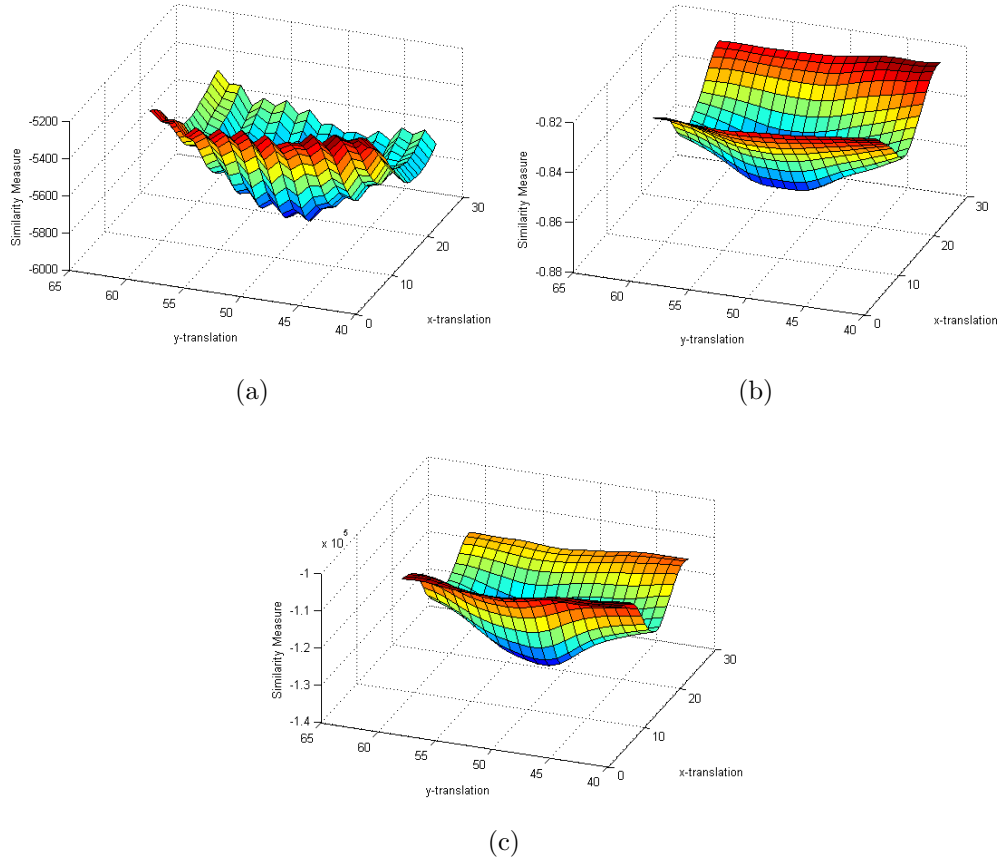


Figure 6.5: Similarity value vs xy-translations for a) DT-CWT, b) FSIM, and c) RWT similarity measures. Images that are used to calculate the measures are given in Figure 6.3.

In Figure 6.6, $x[n]$ is the input image, $x_1[n]$, $x_2[n]$, $x_3[n]$, $x_4[n]$ are the approximation, horizontal, vertical and diagonal wavelet coefficients, respectively. Filters $h[n]$ and $g[n]$ are first applied vertically and then horizontally.

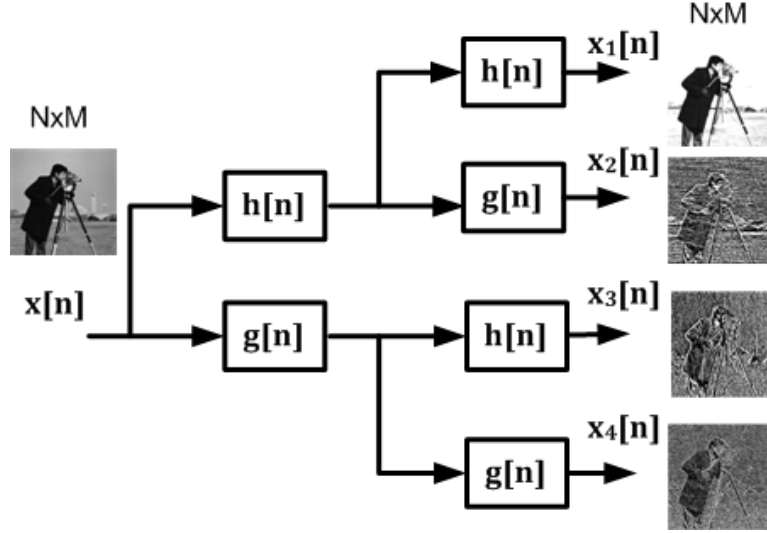


Figure 6.6: 1-Level redundant wavelet transform.

Based on the experimental results, we decide to use redundant (undecimated) wavelet transform to incorporate spatial information into the cost function. In redundant wavelet transform down-sampling/up-sampling stages of the standard discrete wavelet transform are removed and each output has dimensions of the input image. We use the amplitudes and angles of the vertical, horizontal and diagonal detail images after 1-level redundant wavelet transform. Edges produce high amplitude wavelet coefficients across the scales. Whenever there is an edge in the regular image or the IR image wavelet coefficients will be higher than other wavelet coefficients corresponding to flat areas. Both images will have high amplitude wavelet coefficients across the scales. Edges in an IR image may not be as sharp as the regular image. As a result, the gradient values of the regular image around an edge may be significantly different from the gradient values of the IR image. Wavelet coefficients across scales will be prevalent in both IR and regular images and they are closer in value compared to gradient values. Gradients are computed only horizontally and vertically in [120]. On the other hand, wavelets have diagonal bands which contribute to the cost function and this results in smoother cost function than the gradient measure. The wavelet measure is defined as follows:

$$WM(I_V, I_R) = \sum_{i,j} \gamma \times \min(|W(I_V(i))|, |W(I_R(j))|), \quad (6.6)$$

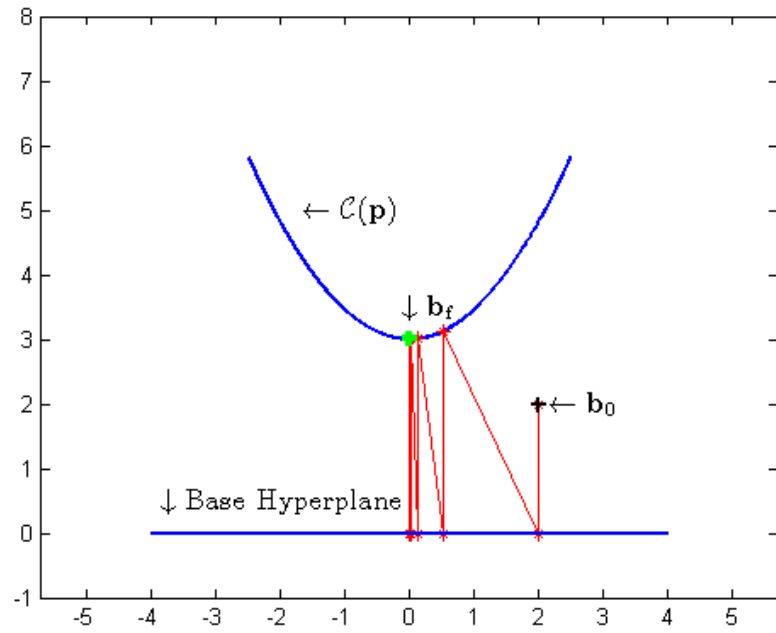
where $|W(I(i))| = [H_I(i), V_I(i), D_I(i)]$ is a vector that consists of the horizontal, vertical and diagonal wavelet coefficients for the i^{th} pixel of the image, and γ is a scaling factor:

$$\gamma = \frac{\cos \left(2 \cos^{-1} \left(\frac{W(I_v(i)) \cdot W(I_r(j))}{|W(I_v(i))| |W(I_r(j))|} \right) \right) + 1}{2}. \quad (6.7)$$

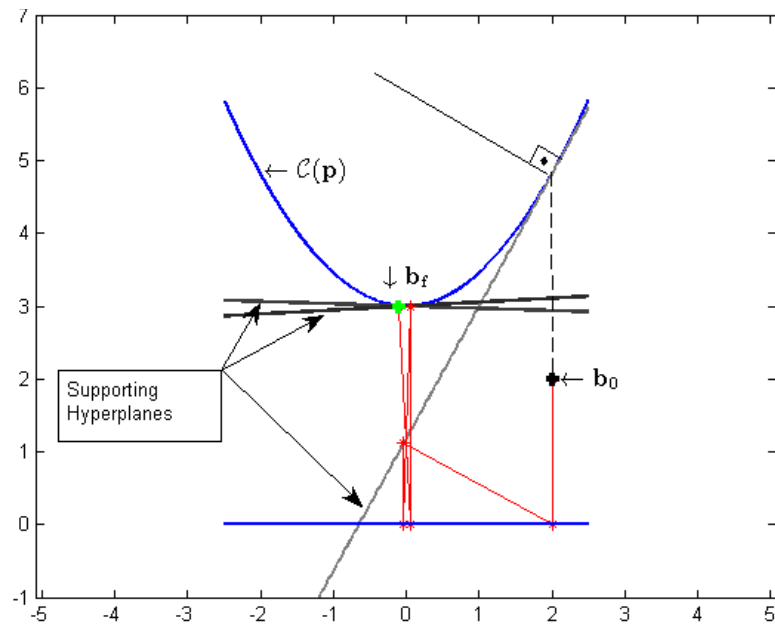
The scaling function, γ , emphasizes small (close to zero) and large (close to π) angles and it is the generalized version of the scaling function introduced in [120] for gradients.

6.3.2 Optimization Algorithm

We use the method introduced in [127] to minimize the negative of the cost function. This method can be used to find the global minimum of convex functions by applying successive projections onto the tangent hyperplanes of the cost function. If the projection operator onto the cost function has a closed form this minimization method can be implemented as shown in Figure 6.7(a), which depicts the one-dimensional optimization method. The dimension of the parameter set is increased (lifted) by one and the algorithm starts at the initial point $\mathbf{b}_0 = [\mathbf{p}_0 \ c_0] = [t_{x0} \ t_{y0} \ c_0]$. The last element (c_0) added to the parameter set will carry the value of the cost function during projections. The initial point is first projected onto the “Base Hyperplane” and then onto the cost function. The method is derived from the widely used “projection onto the convex sets” [128, 129] algorithm. Successive projections onto two convex sets (the cost function $\mathcal{C}$ and the “Base Hyperplane”), converge to their intersection or the closest vectors in each set. Projections should oscillate between the minimum of the cost function and the base hyperplane as shown in Figure 6.7(a).



(a)



(b)

Figure 6.7: a) Minimization of a convex function by projections when closed form expression of the projection operator is available. b) Minimization of a convex function by projections onto supporting hyperplanes of the cost function.

When the closed form solution to the projection onto the cost function is not available projections onto the supporting hyperplanes can be used. This method of minimization is displayed in Figure 6.7(b). First a line is drawn from the initial point to the cost function. This line intersects the cost function at $C_0 = \mathcal{C}(\mathbf{p}_0)$, where the transformation operator T is obtained from the first two parameters of the initial point $\mathbf{b}_0 = [t_{x0}, t_{y0}, c_0]$. Then a supporting hyperplane formulated by $\mathbf{D}^T \times \mathbf{b} = w$ is defined at this point with parameters $D^T = [\nabla \mathcal{C}(\mathbf{p}_0) \quad -1]$ and $w = D^T \times [\mathbf{p}_0 \quad c_0]$. The gradient of the cost function is calculated with respect to the parameter using the method in Equation 6.10. Implementation details and proposed improvements of the optimization method are provided in Appendix B. We need to also determine the minimum value that the cost function can achieve to determine the parameters of the base hyperplane. After finding the minimum value we scale the cost function to a predetermined range so that it can be used in the projection based optimization algorithm. The scaling value can be determined as follows:

$$\beta = \frac{\alpha}{\min(|\mathcal{C}(I_V, T(I_V, \mathbf{p}))|, |\mathcal{C}(I_R, T(I_R, \mathbf{p}))|)}, \quad (6.8)$$

where $\mathbf{p}$ contains the identity transformation parameters and α is an experimentally determined positive constant that results in fast convergence depending on the range of transformation parameters. Using this scaling factor the cost function will be used in the optimization in the following form:

$$\mathcal{C}(\mathbf{p}) = \beta \times \mathcal{C}(I_V, T(I_R, \mathbf{p})). \quad (6.9)$$

The proposed cost function is not necessarily convex, it can be defined as quasi-convex for some of the parameters. Random-restart technique can be used to rerun the optimization algorithm at random initial positions to find the global minimum if necessary.

We use numerical methods to estimate the derivative of the cost function ($\mathcal{C}$) [130]. The first method uses the standard finite difference derivative estimation and requires two evaluations of the cost function for each transformation parameter ($\mathbf{p}_k$):

$$\frac{\partial \mathcal{C}(\mathbf{p}_k)}{\partial \mathbf{p}[i]} = \frac{[\mathcal{C}(\mathbf{p}_k + \Delta e_i)] - [\mathcal{C}(\mathbf{p}_k - \Delta e_i)]}{2\Delta e_i}, \quad (6.10)$$

where Δe_i is the perturbation applied to the i^{th} parameter, and $\mathbf{p}_k$ is the transformation parameter set at the k^{th} iteration.

The second method is called the “simultaneous perturbation” and requires a total of two evaluations of the cost function:

$$\frac{\partial \mathcal{C}(\mathbf{p}_k)}{\partial \mathbf{p}[i]} = \frac{[\mathcal{C}(\mathbf{p}_k + c_k \Delta_k)] - [\mathcal{C}(\mathbf{p}_k - c_k \Delta_k)]}{2c_k \Delta_k[i]}, \quad (6.11)$$

where Δ_k is the random perturbation vector and c_k is the update parameter. Although the first method is a coarse approximation to the true derivative, it is shown in [130] that it has comparable performance to the analytical derivative.

6.4 Scale and Rotation Estimation

We modify the image registration method introduced in [131] to work with infrared and visible range images. In this method the main idea used is the fact that scale and rotation can be represented as translations in log-polar domain. The translations in Cartesian domain produce phase difference in Fourier domain. Therefore, we convert images to Fourier domain and Fourier magnitudes are converted to log-polar domain to be able to represent scale and rotation as translation without the effects of Cartesian domain translations. Then, phase correlation is used to find translations in log-polar domain.

In our method we replace the last step to determine scale and rotation for multi-modal images. Instead of phase correlation we use the mutual information between the log-polar images and use a genetic algorithm based optimization to find the rotation and scale parameters.

The image coordinates in Cartesian space (x, y) can be converted to log-polar domain coordinates (ρ, θ) as follows:

$$\rho = \sqrt{(x - x_c)^2 + (y - y_c)^2}, \quad (6.12)$$

$$\theta = \arctan\left(\frac{y - y_c}{x - x_c}\right), \quad (6.13)$$

where (x_c, y_c) is the center of the image in Cartesian coordinates. Figure 6.8 shows an image, its scaled and rotated version and their log-polar domain counterparts.

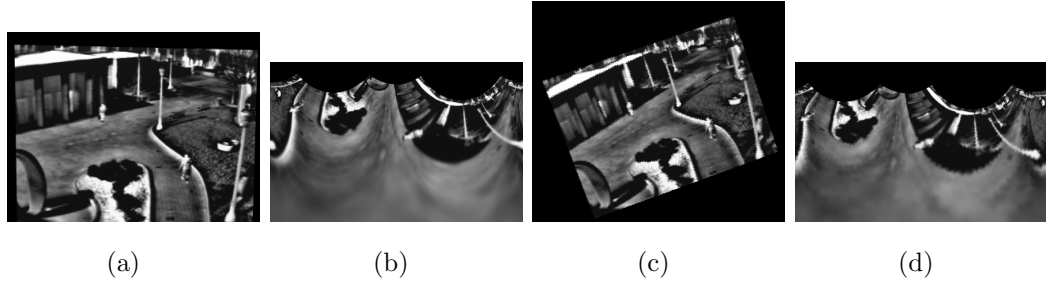


Figure 6.8: a) Reference image (from [3]), b) log-polar transforms of reference image, c) scaled/rotated image, d) log-polar transform of scaled/rotated image.

The cost function for scale and rotation uses log-polar transforms in the Fourier domain together with mutual information to recover scale and orientation. This cost function is defined as follows:

$$\mathcal{C}_{sr}(I_V, T(I_R, \mathbf{p})) = \text{MI}_{sr}(\text{LPTC}(I_V), \text{LPTC}(T(I_R, \mathbf{p}))), \quad (6.14)$$

where MI_{sr} defines the mutual information similarity functions for the scale and rotation parameters, LPTC is the log-polar transform of the FFT magnitudes of the images defined for an image I as:

$$\text{LPTC}(I) = \text{LP}(\text{HP}(|\text{FFT}(I)|)), \quad (6.15)$$

where LP is the log-polar transform, FFT is the fast Fourier transform and HP is a high-pass filter.

Figure 6.9 shows visible and IR images, their FFT magnitudes and log-polar transforms. When unimodal images are registered phase correlation can be used to estimate rotation and scale as translations in log-polar space. In the multi-modal images case, phase-correlation does not work, therefore, we propose to use mutual information.

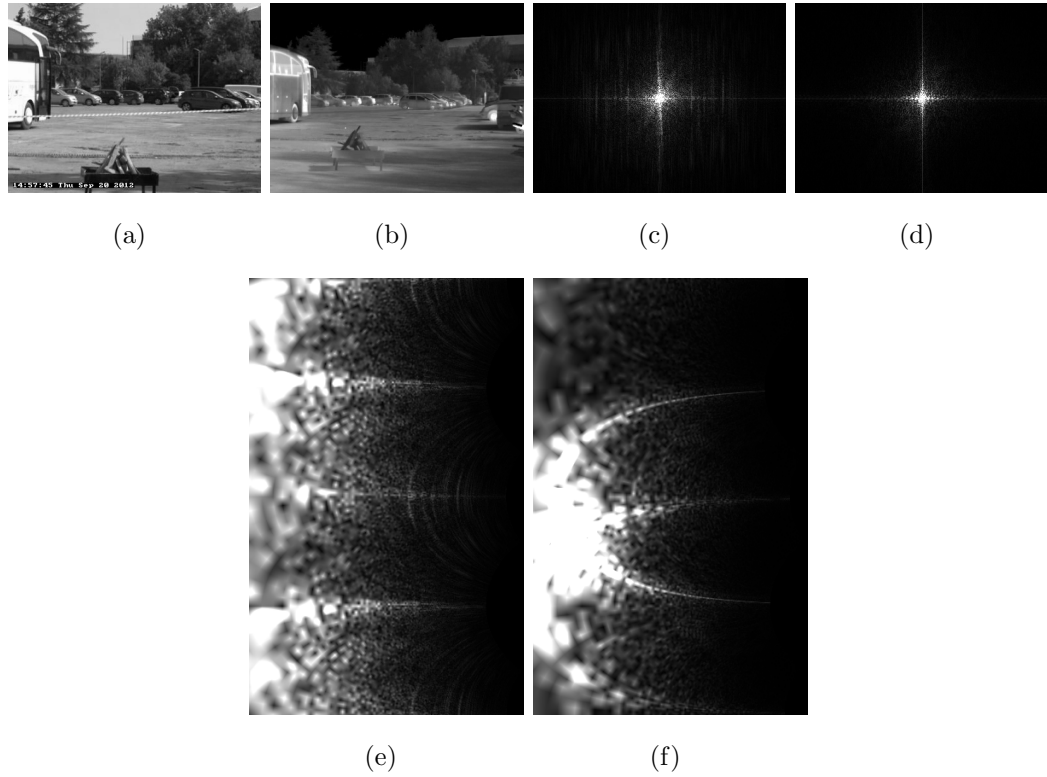


Figure 6.9: a) Visible range image b) IR image, c) FFT of visible range image, d) FFT of IR images, e) log-polar transform of FFT of visible image, f) log-polar transform of FFT of IR image.

Figure 6.10 shows the negative of the mutual information value vs scale and rotations. The cost function is not smooth for this case and that is the reason why we use a genetic algorithm [132] to minimize the cost function. The proposed method can extract scale and rotation values from multi-modal images even when they are translated in Cartesian coordinates. However, there might be some problematic results when the translations are so large that the intersecting regions of images does not correspond to the same regions even if they are at the same scale and rotation. We can overcome this problem by just marking a single point on each image that corresponds to the same location. The location of the points does not have to be precise, they just provide a coarse reference point to find the intersecting regions of images.

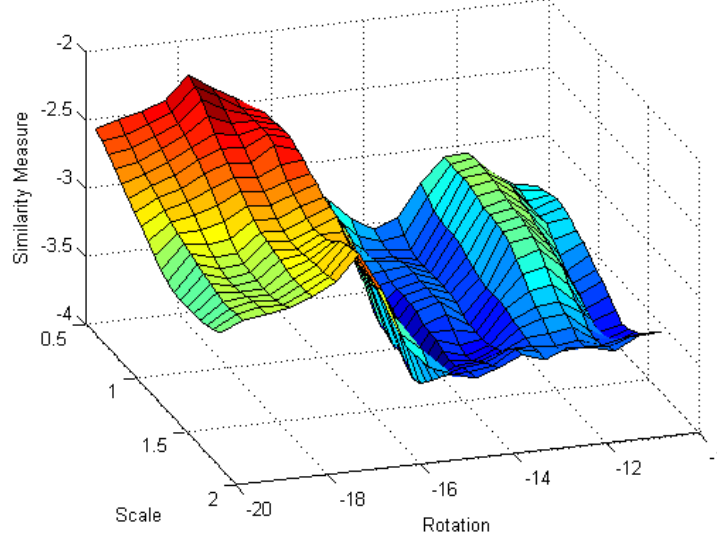


Figure 6.10: Mutual information cost value vs scale and rotations.

6.5 Experimental Results

We first compare the proposed similarity measure and the optimization method for finding translations with the gradient descent (GD) and mutual information based methods. Then, we compare the full algorithm with state of the art image registration methods. Gradient descent is the standard method used in the minimization of convex functions:

$$\mathbf{p} = \mathbf{p} - \mu \nabla \mathcal{C}(\mathbf{p}), \quad (6.16)$$

where $\mathbf{p}$ is the parameter vector and μ is the update parameter. Update parameter needs to be determined at each iteration for every transformation parameter. The determination of this parameter is usually done by trial and error. The most important advantage of the projection based optimization algorithm is that it does not need an update parameter. The pseudo-code of the projection based optimization method is given in Algorithm 2. The only parameter to be defined is the threshold ϵ that stops the algorithm when the difference between successive cost values becomes small.

Algorithm 2 Projection based optimization algorithm

Initial parameters:

$i=1$, ϵ =Small Constant, $b_{02} = [p_{02} \ c_{02}]$ and $b_{min} = b_{02}$

C_{min} =DBL_MAX //Max double value

do

$\mathbf{b}_{i0} = \mathbf{b}_{(i-1)2}$

Project onto base hyperplane. Determine $\mathbf{b}_{i1}$

Find cost at $\mathbf{p}_{i1} : \mathcal{C}(\mathbf{p}_{i1})$

Update C_{min} and $\mathbf{b}_{min}$

if $\mathcal{C}(\mathbf{b}_{i1}) > C_{min}$ **then**

$$\mathbf{b}_{i2} = \frac{|\mathcal{C}(\mathbf{b}_{i1})|\mathbf{b}_{i1} + |\mathcal{C}(\mathbf{b}_{min})|\mathbf{b}_{min}}{|\mathcal{C}(\mathbf{b}_{i1})| + |\mathcal{C}(\mathbf{b}_{min})|}$$

else

Find gradient at $\mathbf{p}_{i1} : \nabla \mathcal{C}(\mathbf{p}_{i1})$

Calculate supporting hyperplane parameters:

$$\mathbf{D}^T \times \mathbf{b} = w$$

$$\mathbf{D}^T = [\nabla \mathcal{C}(\mathbf{p}_{i1}) \ -1]$$

$$w = \mathbf{D}^T \times [\mathbf{p}_{i1} \ c_{i1}]$$

Project onto supporting hyperplane. Determine $\mathbf{b}_{i2}$

Find cost at $\mathbf{p}_{i2} : \mathcal{C}(\mathbf{p}_{i2})$

end if

while $(|\mathcal{C}(\mathbf{p}_{i2}) - C_{min}| > \epsilon)$

In the first experiment, we register visible range and IR images that only differ in x-y translations. The images, joint histograms and the cost function vs translations are shown in Figure 6.11. The evolution of the x-y translation and cost values versus iterations are shown in Figure 6.12 for the proposed and gradient descent based algorithms. We observe that the our method reaches convergence much faster than the gradient descent based method. Table 6.5 shows the actual values of transformation parameters (given in parentheses) and the values obtained from both algorithms.

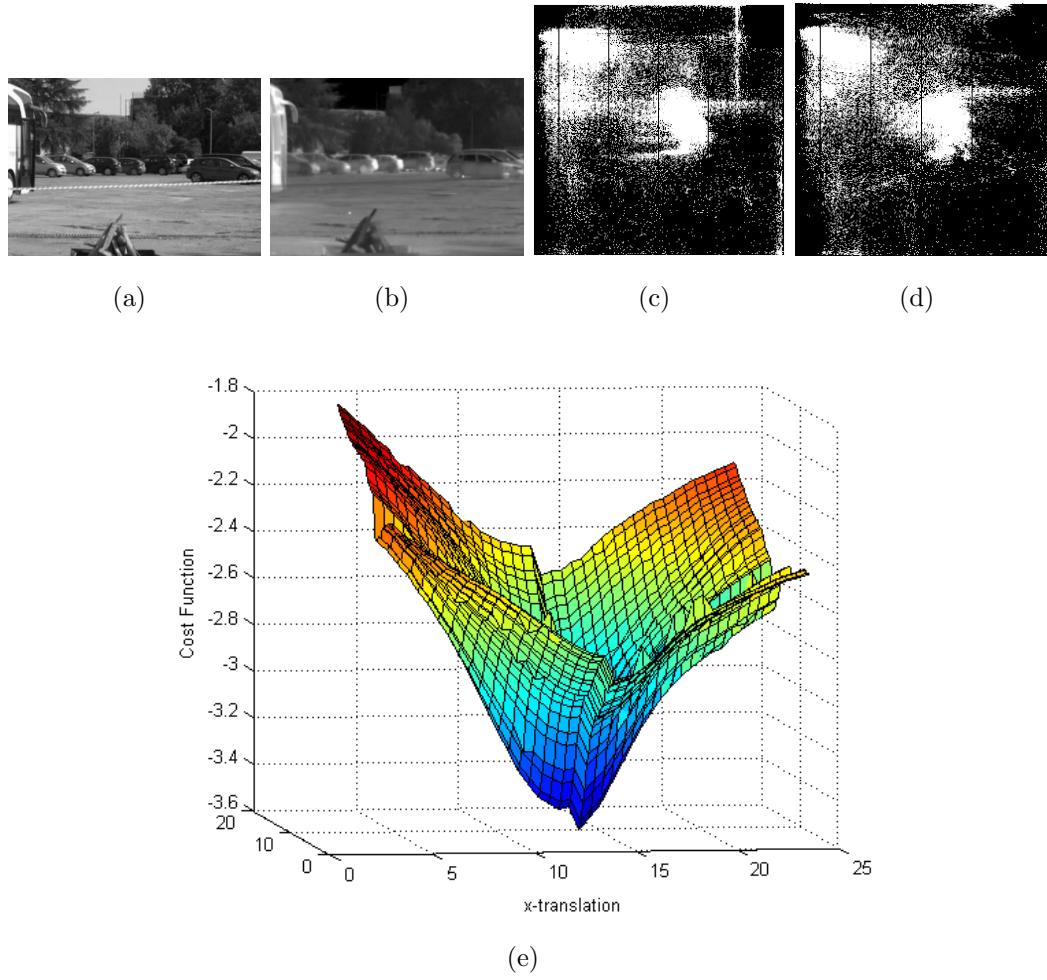


Figure 6.11: a) Visible range image b) IR image, c) joint histogram for unregistered images, d) joint histogram for registered images, e) cost function vs translations.

Table 6.1: Registration results for the first experiment comparing gradient descent proposed methods. Here t_x and t_y represent x-y translations.

	Proposed	GD Based
t_x (14)	13.84	13.66
t_y (10)	9.80	9.59

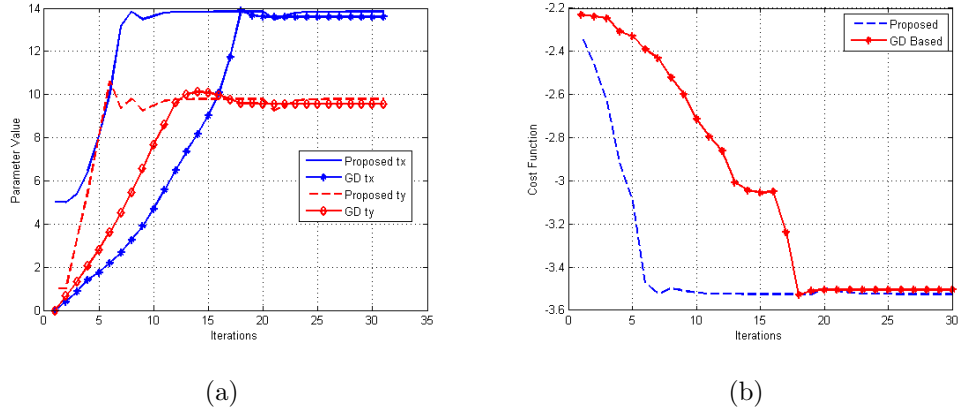


Figure 6.12: a) Evolution of transformation parameters vs iterations, b) evolution of the cost function vs iterations.

In the second part of the experiment, we compare our method with MATLAB’s multi-modal image registration functions which are based on the Insight Segmentation and Registration Toolkit (ITK) [133]. ITK method uses mutual information as the similarity measure and gradient descent or evolutionary algorithms as the optimization method. We also compare the results of manual registration which is performed by manually selecting matching points in both images and finding a similarity transform using random sample consensus (RANSAC) [134].

Table 6.2: Registration results for the second experiment.

Error is defined as: $[\Delta s \times 10 + \Delta \phi \times 10 + \Delta t_x + \Delta t_y]$.

Δ represents the difference between the true value of the parameter and the optimization result.

	Image Set 1				
	s	ϕ (degrees)	t_x	t_y	<i>error</i>
Ground Truth	1.250	-15.00	-75.84	-108.66	0
Proposed	1.248	-14.61	-74.42	-105.00	9
Manual	1.259	-14.68	-78.23	-106.55	7.7
ITK	1.242	-15.12	-74.13	-105.59	6
	Image Set 2				
	s	ϕ (degrees)	t_x	t_y	<i>error</i>

Ground Truth	1.490	-0.32	-7.68	-48.35	0
Proposed	1.471	-0.34	-6.00	-48.00	2.42
Manual	1.490	-0.32	-7.68	-48.35	0
ITK	1.418	-0.93	-4.54	-41.26	17.5
Image Set 3					
	s	ϕ (degrees)	t_x	t_y	<i>error</i>
Ground Truth	0.830	0.00	-13.00	-4.00	0
Proposed	0.826	0.00	-13.36	-4.91	1.31
Manual	0.843	0.46	-16.72	-8.88	13.31
ITK	0.851	-0.65	-18.37	-7.63	15.71
Image Set 4					
	s	ϕ (degrees)	t_x	t_y	<i>error</i>
Ground Truth	1.540	0.00	-9.00	-7.00	0
Proposed	1.612	-0.46	-7.03	-4.02	10.27
Manual	1.527	-0.39	0.43	2.88	23.34
ITK	1.550	-0.17	0.23	2.45	20.48
Image Set 5a					
	s	θ	t_x	t_y	<i>error</i>
Ground Truth	1.400	10.00	-61.03	-61.00	0
Proposed	1.399	9.91	-60.73	-60.62	1.59
Manual	1.400	9.94	-60.44	-61.09	1.28
ITK	1.401	10.02	-60.27	-60.17	1.8
Image Set 5b					
	s	θ	t_x	t_y	<i>error</i>
Ground Truth	0.650	-4.00	-11.66	-11.86	0
Proposed	0.646	-4.07	-11.31	-11.70	1.25
Manual	0.647	-3.66	-11.55	-10.65	4.75
ITK	0.784	1.22	-9.14	-14.03	58.23
Image Set 6a					
	s	θ	t_x	t_y	<i>error</i>
Ground Truth	1.300	-11.00	-110.99	-127.64	0

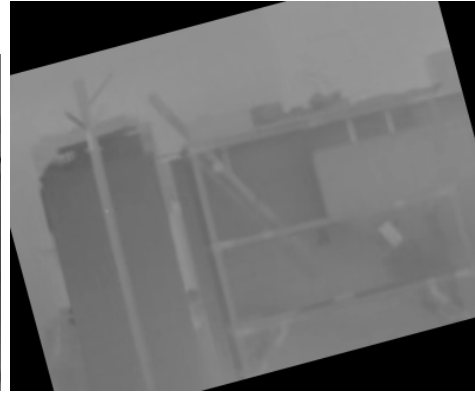
Proposed	1.298	-10.83	-109.98	-125.83	4.54
Manual	1.305	-10.94	-113.26	-128.05	3.33
ITK	1.300	-11.01	-110.45	-126.34	1.94
	Image Set 6b				
	s	θ	t_x	t_y	$error$
Ground Truth	0.720	-18.00	-96.54	-110.78	0
Proposed	0.721	-16.44	-95.66	-107.71	19.56
Manual	0.718	-18.28	-95.98	-111.40	4
ITK	0.720	-18.02	-102.43	-111.09	6.4

The methods are compared on four different image sets. Image Set 2 consists of images directly obtained from different cameras. The results of manual registration are used as the ground truth values for this image set. The other images are obtained by transforming previously registered images to perform controlled experiments. Images Sets 3 and 4 are obtained from [3]. Image Sets 5 and 6 are hyperspectral satellite images [4]. The results of the experiments are shown in Table 6.2. The proposed method surpasses ITK and has similar performance to manual registration. The performance of ITK method is very dependent on the initial transformation parameters. We therefore used initial transformation that is in the vicinity of the correct transformation, but even then it failed on some of the datasets.

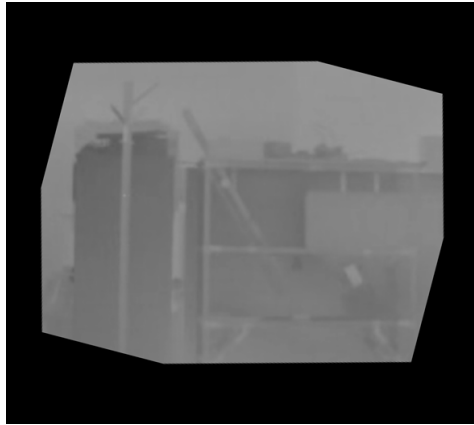
Figure 6.13 shows the images in various stages of registration for Image Set 1.



(a)



(b)



(c)



(d)

Figure 6.13: Registration results for Image Set 1: a) visible Image, b) IR image, c) scaled and rotate IR image, translated IR image.

Chapter 7

Conclusion

We proposed many image and video processing algorithms for wildfire surveillance applications. The algorithms range from online learning to feature extraction and function optimization based methods.

An entropy functional based online classifier fusion (ECF) method is proposed for video processing applications. In this framework, the weights of the combiner are obtained using non-orthogonal e-projections onto convex sets representing each classifier. This method is tested using a real-time wildfire detection algorithm.

We studied online learning algorithms from a generalized update and performed extensive experiments to validate their performance. We developed stochastic gradient based algorithms and a mixture of experts algorithm. We test the algorithms with synthetic and real machine learning datasets. We also developed a real-time wildfire detection method that can be used to detect wildfire smoke from aerial video sequences without using motion information. From the results we see that the performance of different algorithms depends on the application.

We investigated the applications of local binary patterns with random hyperplanes and deep neural networks for real-time dynamic texture recognition. We use random sampling and dimensionality reduction to significantly speed-up the feature extraction and classification process. We observed that using random hyperplanes can even increase the recognition performance compared to the full-sized feature vectors. The results of experiments on DynTex dynamic texture database and infrared flame detection database prove the real-time performance of the method.

We developed a real-time wildfire detection algorithm to reduce the scanning period of conventional stop-and-detect methods by using continuously moving cameras. We investigated the performance of SURF features in low-quality images recorded with wildfire surveillance cameras. We showed that MI based method can provide robustness against low quality images during the registration process. We proposed a wildfire detection method that can exploit robust features to characterize smoke behaviour. Experimental results for feature extraction and real-time tests prove the performance of the proposed algorithm.

We developed a multi-modal image registration method to register infrared and visible images. The proposed method can be used in surveillance applications to match the detection results from different cameras. We obtained a similarity transform that can account for scale, rotations and translations. We proposed a mutual information based translation invariant method to recover scale and rotation. We developed a similarity measure based on mutual information and redundant wavelet transform to find translation parameters. We experimentally observed that redundant wavelet transform produces better results than gradient based methods. We proposed a projection based minimization method that is shown to be more robust and flexible than gradient descent algorithms since it does not require an update parameter to reach the global minima. The developed method is compared with state of the art systems and has shown very good performance.

Bibliography

- [1] R. Péteri, S. Fazekas, and M. J. Huiskes, “DynTex : a Comprehensive Database of Dynamic Textures,” *Pattern Recognition Letters*, vol. 31, no. 12, pp. 1627 – 1632, 2010.
- [2] N. Goyette, P. Jodoin, F. Porikli, J. Konrad, and P. Ishwar, “Changedetec-tion.net: A new change detection benchmark dataset,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Work-shops (CVPRW)*, pp. 1–8, June 2012.
- [3] J. W. Davis and V. Sharma, “Background-subtraction using contour-based fusion of thermal and visible imagery,” *Computer Vision and Image Un-derstanding*, vol. 106, no. 23, pp. 162 – 182, 2007.
- [4] S. Yahyanejad and B. Rinner, “A fast and mobile system for registration of low-altitude visual and thermal aerial images using multiple small-scale UAVs,” *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 104, no. 1, pp. 189 – 202, 2015.
- [5] S. Suri and P. Reinartz, “Mutual-Information-Based Registration of TerraSAR-X and Ikonos Imagery in Urban Areas,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 48, pp. 939–949, Feb 2010.
- [6] J. Kern and M. Pattichis, “Robust Multispectral Image Registration Us-ing Mutual-Information Models,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 45, pp. 1494–1505, May 2007.
- [7] S. Verstockt, A. Vanoosthuyse, S. Hoecke, P. Lambert, and R. Walle, “Multi-sensor fire detection by fusing visual and non-visual flame features,”

- in *Image and Signal Processing*, vol. 6134 of *Lecture Notes in Computer Science*, pp. 333–341, Springer Berlin Heidelberg, 2010.
- [8] A. E. Cetin, K. Dimitropoulos, B. Gouverneur, N. Grammalidis, O. Gunay, Y. H. Habiboglu, B. U. Töreyn, and S. Verstockt, “Video fire detection - review,” *Digital Signal Processing*, vol. 23, no. 6, pp. 1827–1843, 2013.
 - [9] O. Gunay, B. U. Toreyin, K. Kose, and A. E. Cetin, “Entropy-Functional-Based Online Adaptive Decision Fusion Framework With Application to Wildfire Detection in Video,” *IEEE Transactions on Image Processing*, vol. 21, no. 5, pp. 2853–2865, 2012.
 - [10] B. U. Töreyn, *Fire Detection Algorithms Using Multimodal Signal and Image Analysis*. PhD thesis, Bilkent University, 2009.
 - [11] S. Soatto, G. Doretto, and Y. N. Wu, “Dynamic textures,” in *Proc. of IEEE International Conference on Computer Vision (ICCV)*, vol. 2, pp. 439–446 vol.2, 2001.
 - [12] D. Chetverikov and R. Pteri, “A brief survey of dynamic texture description and recognition,” in *Computer Recognition Systems*, vol. 30 of *Advances in Soft Computing*, pp. 17–26, Springer Berlin Heidelberg, 2005.
 - [13] O. Günay, K. Taşdemir, B. U. Töreyn, and A. E. Çetin, “Video based wildfire detection at night,” *Fire Safety Journal*, vol. 44, no. 6, pp. 860–868, 2009.
 - [14] S. Theodoridis, K. Slavakis, and I. Yamada, “Adaptive Learning in a World of Projections,” *IEEE Signal Processing Magazine*, vol. 28, no. 1, pp. 97–123, 2011.
 - [15] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer, “Online Passive-Aggressive Algorithms,” *The Journal of Machine Learning Research*, vol. 7, pp. 551–585, 2006.
 - [16] W. Liu, J. C. Principe, and S. Haykin, *Kernel Adaptive Filtering: A Comprehensive Introduction*. Wiley Publishing, 1st ed., 2010.

- [17] J. Kivinen and M. K. Warmuth, “Exponentiated Gradient versus Gradient Descent for Linear Predictors,” *Information and Computation*, vol. 132, no. 1, pp. 1 – 63, 1997.
- [18] M. A. Donmez, H. A. Inan, and S. S. Kozat, “Adaptive mixture methods based on Bregman divergences,” *Digital Signal Processing*, vol. 23, no. 1, pp. 86 – 97, 2013.
- [19] T. Ojala, M. Pietikainen, and D. Harwood, “Performance evaluation of texture measures with classification based on kullback discrimination of distributions,” in *Proc. of IAPR International Conference on Pattern Recognition (ICPR)*, vol. 1, pp. 582–585 vol.1, Oct 1994.
- [20] G. Zhao and M. Pietikainen, “Dynamic texture recognition using local binary patterns with an application to facial expressions,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, pp. 915–928, June 2007.
- [21] D. Lowe, “Distinctive image features from scale-invariant keypoints,” *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.
- [22] W. M. W. III, P. A. Viola, and H. Atsumi, “Multi-modal volume registration by maximization of mutual information,” *Medical Image Analysis*, vol. 1, no. 1, pp. 35–51, 1996.
- [23] Y. S. Kim, J. H. Lee, and J. B. Ra, “Multi-sensor image registration based on intensity and edge orientation information,” *Pattern Recognition*, vol. 41, pp. 3356–3365, November 2008.
- [24] R. Polikar, “Ensemble based systems in decision making,” *IEEE Circuits and Systems Magazine*, vol. 6, pp. 21–45, Third 2006.
- [25] O. Gunay, B. U. Toreyin, and A. E. Cetin, “Online adaptive decision fusion framework based on projections onto convex sets with application to wildfire detection in video,” *Optical Engineering*, vol. 50, no. 7, pp. 077202–077202–12, 2011.

- [26] B. Toreyin and A. Cetin, “Wildfire detection using lms based active learning,” in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1461–1464, April 2009.
- [27] L. G. Gubin, B. T. Polyak, and E. V. Raik, “The method of projections for finding the common point of convex sets,” *USSR Computational Mathematics and Mathematical Physics*, vol. 7, no. 6, pp. 1 – 24, 1967.
- [28] D. C. Youla and H. Webb, “Image restoration by the method of convex projections, Part I-Theory,” *IEEE Transactions on Medical Imaging*, vol. MI-I-2, pp. 81–94, 1982.
- [29] A. E. Çetin, “Reconstruction of signals from Fourier transform samples,” *Signal Processing*, vol. 16, pp. 129–148, 1989.
- [30] K. Slavakis, S. Theodoridis, and I. Yamada, “Online Kernel-Based Classification Using Adaptive Projection Algorithms,” *IEEE Transactions on Signal Processing*, vol. 56, pp. 2781–2796, 2008.
- [31] U. Niesen, D. Shah, and G. Wornell, “Adaptive Alternating Minimization Algorithms,” *IEEE Transactions on Information Theory*, vol. 55, no. 3, pp. 1423–1429, 2009.
- [32] I. Yamada and N. Ogura, “Adaptive Projected Subgradient Method for Asymptotic Minimization of Sequence of Nonnegative Convex Functions,” *Numerical Functional Analysis and Optimization*, vol. 25, no. 7, pp. 593–617, 2005.
- [33] K. Slavakis, Y. I., and O. N., “The adaptive projected subgradient method over the fixed point set of strongly attracting nonexpansive mappings,” *Numerical Functional Analysis and Optimization*, vol. 27, no. 7, pp. 905–930, 2006.
- [34] A. E. Çetin and R. Ansari, “Signal recovery from wavelet transform maxima,” *IEEE Transactions on Signal Processing*, vol. 42-1, pp. 194–196, 1994.
- [35] P. Combettes, “The foundations of set theoretic estimation,” *Proceedings of the IEEE*, vol. 81, no. 2, pp. 182–208, 1993.

- [36] K. Kose, O. Gunay, and A. E. Cetin, “Compressive Sensing Using the Modified Entropy Functional,” *Digital Signal Processing*, vol. 24, pp. 63–70, Jan. 2014.
- [37] L. Bregman, “The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming,” *USSR Computational Mathematics and Mathematical Physics*, vol. 7, no. 3, pp. 200 – 217, 1967.
- [38] Y. Censor and A. Lent, “Optimization of “log x” entropy over linear equality constraints,” *SIAM Journal on Control and Optimization*, vol. 25, pp. 921–933, 1987.
- [39] G. T. Herman, “Image Reconstruction From Projections,” *Real-Time Imaging*, vol. 1, no. 1, pp. 3–18, 1995.
- [40] Y. H. Habiboglu, O. Gunay, and A. E. Cetin, “Real-time wildfire detection using correlation descriptors,” in *Proc. of European Signal Processing Conference*, 2011.
- [41] FIRESENSE, “Fire Detection and Management through a Multi-Sensor Network for the Protection of Cultural Heritage Areas from the Risk of Fire and Extreme Weather Conditions, FP7-ENV-2009-1244088-FIRESENSE,” 2009. <http://www.firesense.eu>.
- [42] P. Guillemant and J. Vicente, “Real-time identification of smoke images by clustering motions on a fractal curve with a temporal embedding method,” *Optical Engineering*, vol. 40, no. 4, pp. 554–563, 2001.
- [43] J. Vicente and P. Guillemant, “An image processing technique for automatically detecting forest fire,” *International Journal of Thermal Sciences*, vol. 41, no. 12, pp. 1113 – 1120, 2002.
- [44] F. Gomez-Rodriguez, B. C. Arrue, and A. Ollero, “Smoke monitoring and measurement using image processing: application to forest fires,” in *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series* (F. A. Sadjadi, ed.), vol. 5094, pp. 404–411, Sept. 2003.

- [45] L. Qinjuan and H. Ning, “Effective dynamic object detecting for video-based forest fire smog recognition,” in *Proc. of 2nd International Congress on Image and Signal Processing CISP*, 2009.
- [46] A. C. Singer and M. Feder, “Universal linear prediction by model order weighting,” *IEEE Transactions on Signal Processing*, vol. 47-10, pp. 2685–2699, 1999.
- [47] O. Dekel, J. Keshet, and Y. Singer, “An Online Algorithm for Hierarchical Phoneme Classification,” in *Machine Learning for Multimodal Interaction* (S. Bengio and H. Bourlard, eds.), vol. 3361 of *Lecture Notes in Computer Science*, pp. 146–158, Springer Berlin Heidelberg, 2005.
- [48] B. Li and S. C. H. Hoi, “Online Portfolio Selection: A Survey,” *ACM Comput. Surv.*, vol. 46, pp. 35:1–35:36, Jan. 2014.
- [49] D. Grangier and S. Bengio, “A Discriminative Kernel-Based Approach to Rank Images from Text Queries,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, pp. 1371–1384, Aug 2008.
- [50] L. Zhang and L. van der Maaten, “Structure Preserving Object Tracking,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1838–1845, June 2013.
- [51] F. Rosenblatt, “The Perception: A Probabilistic Model for Information Storage and Organization in the Brain,” *Neurocomputing: Foundations of Research*, pp. 89–114, 1988.
- [52] Y. Freund and R. E. Schapire, “Large Margin Classification Using the Perceptron Algorithm,” *Machine Learning*, vol. 37, pp. 277–296, Dec. 1999.
- [53] L. Wang, H.-B. Ji, and Y. Jin, “Fuzzy Passive-Aggressive classification: A robust and efficient algorithm for online classification problems,” *Information Sciences*, vol. 220, no. 1, pp. 46 – 63, 2013.
- [54] Y. Li and P. M. Long, “The Relaxed Online Maximum Margin Algorithm,” *Machine Learning*, vol. 46, pp. 361–387, Jan. 2002.

- [55] K. Crammer and Y. Singer, “Ultraconservative Online Algorithms for Multiclass Problems,” *Journal of Machine Learning Research*, vol. 3, p. 2003, 2001.
- [56] C. Gentile, “A New Approximate Maximal Margin Classification Algorithm,” *Journal of Machine Learning Research*, vol. 2, pp. 213–242, Mar. 2002.
- [57] C. Cortes and V. Vapnik, “Support-Vector Networks,” *Machine Learning*, vol. 20, pp. 273–297, Sept. 1995.
- [58] J. Kivinen, A. J. Smola, and R. C. Williamson, “Online Learning with Kernels,” *IEEE Transactions on Signal Processing*, vol. 52, pp. 2165–2176, August 2004.
- [59] K. Crammer, J. Kandola, R. Holloway, and Y. Singer, “Online Classification on a Budget,” in *Advances in Neural Information Processing Systems*, pp. 20–33, MIT Press, 2003.
- [60] O. Dekel, S. Shalev-shwartz, and Y. Singer, “The Forgetron: A kernel-based perceptron on a fixed budget,” in *Proc. of Advances in Neural Information Processing Systems*, pp. 259–266, MIT Press, 2005.
- [61] Z. Wang and S. Vucetic, “Online Passive-Aggressive Algorithms on a Budget,” in *Proc. of International Conference on Artificial Intelligence and Statistics (AISTATS)*, vol. 9, pp. 908–915, 2010.
- [62] F. Orabona, J. Keshet, and B. Caputo, “The Projectron: A Bounded Kernel-based Perceptron,” in *Proc. of International Conference on Machine Learning (ICML)*, (New York, NY, USA), pp. 720–727, ACM, 2008.
- [63] N. Pavlidis, D. Tasoulis, N. Adams, and D. Hand, “ λ -Perceptron: An adaptive classifier for data streams,” *Pattern Recognition*, vol. 44, no. 1, pp. 78 – 96, 2011.
- [64] T. Hofmann, B. Schölkopf, and A. J. Smola, “Kernel Methods in Machine Learning,” *Annals of Statistics*, vol. 36, pp. 1171–1220, 2008.

- [65] F. Orabona, “Dimension-Free Exponentiated Gradient,” in *Neural Information Processing Systems (NIPS)*, pp. 1806–1814, 2013.
- [66] S. Shalev-Shwartz, “Online Learning and Online Convex Optimization,” *Foundations and Trends in Machine Learning*, vol. 4, pp. 107–194, Feb. 2012.
- [67] P. F. Felzenszwalb and D. P. Huttenlocher, “Efficient Graph-Based Image Segmentation,” *International Journal of Computer Vision*, vol. 59, no. 2, pp. 167–181, 2004.
- [68] I. Selesnick, R. Baraniuk, and N. Kingsbury, “The dual-tree complex wavelet transform,” *IEEE Signal Processing Magazine*, vol. 22, pp. 123–151, Nov 2005.
- [69] A. Tahmasbi, F. Saki, and S. B. Shokouhi, “Classification of benign and malignant masses based on Zernike moments,” *Computers in Biology and Medicine*, vol. 41, no. 8, pp. 726–735, 2011.
- [70] F. Orabona, *DOGMA: A MATLAB Toolbox for Online Learning*, 2009. Software available at <http://dogma.sourceforge.net>.
- [71] K. Bache and M. Lichman, “UCI Machine Learning Repository,” 2013. <http://archive.ics.uci.edu/ml>.
- [72] J. Davis and M. Keck, “A two-stage template approach to person detection in thermal imagery,” in *Proc. of IEEE Workshops on Application of Computer Vision (WACV/MOTIONS)*, vol. 1, pp. 364–369, Jan 2005.
- [73] O. Tuzel, F. Porikli, and P. Meer, “Region Covariance: A Fast Descriptor for Detection And Classification,” in *Proc. of European Conference on Computer Vision (ECCV)*, pp. 589–600, 2006.
- [74] G. Zhao and M. Pietikainen, “Local Binary Pattern Descriptors for Dynamic Texture Recognition,” in *Proc. of International Conference on Pattern Recognition (ICPR)*, vol. 2, pp. 211–214, 2006.

- [75] A. Ravichandran, R. Chaudhry, and R. Vidal, “View-invariant dynamic texture recognition using a bag of dynamical systems,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1651–1657, June 2009.
- [76] Z. Lu, W. Xie, J. Pei, and J. Huang, “Dynamic Texture Recognition by Spatio-Temporal Multiresolution Histograms,” in *Proc. of Seventh IEEE Workshops on Application of Computer Vision (WACV/MOTIONS)*, vol. 2, pp. 241–246, Jan 2005.
- [77] K. Derpanis and R. Wildes, “Dynamic texture recognition based on distributions of spacetime oriented structure,” in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 191–198, June 2010.
- [78] B. Ghanem and N. Ahuja, “Phase Based Modelling of Dynamic Textures,” in *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pp. 1–8, Oct 2007.
- [79] Y. Xu, Y. Quan, H. Ling, and H. Ji, “Dynamic texture classification using dynamic fractal analysis,” in *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pp. 1219–1226, Nov 2011.
- [80] J. Chen, G. Zhao, M. Salo, E. Rahtu, and M. Pietikainen, “Automatic dynamic texture segmentation using local descriptors and optical flow,” *IEEE Transactions on Image Processing*, vol. 22, pp. 326–339, Jan 2013.
- [81] J. Chen, G. Zhao, M. Salo, E. Rahtu, and M. Pietikainen, “Automatic dynamic texture segmentation using local descriptors and optical flow,” *IEEE Transactions on Image Processing*, vol. 22, pp. 326–339, Jan 2013.
- [82] Y. Qiao and L. Weng, “Hidden markov model based dynamic texture classification,” *IEEE Signal Processing Letters*, vol. 22, pp. 509–512, April 2014.
- [83] S. Arashloo and J. Kittler, “Dynamic Texture Recognition Using Multiscale Binarized Statistical Image Features,” *IEEE Transactions on Multimedia*, vol. 16, pp. 2099–2109, Dec 2014.

- [84] Y. Qiao, Y. Zhao, and X. Song, “Dynamic Texture Classification Based on Motion Statistical Feature Matrix,” in *Proc. of International Conference on Intelligent Information Hiding and Multimedia Signal Processing (ICIHMSP)*, pp. 535–538, Oct 2013.
- [85] R. Mourya, S. Dubois, O. Alata, and A. Tremeau, “Improving dynamic texture recognition by using a color spatio-temporal decomposition,” in *Proc. of European Signal Processing Conference (EUSIPCO)*, pp. 1–5, Sept 2013.
- [86] J. Ren, X. Jiang, and J. Yuan, “Dynamic texture recognition using enhanced LBP features,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2400–2404, May 2013.
- [87] M. S. Charikar, “Similarity estimation techniques from rounding algorithms,” in *Proc. of Annual ACM Symposium on Theory of Computing (STOC)*, pp. 380–388, 2002.
- [88] A. Andoni and P. Indyk, “Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions,” *Communications of ACM*, vol. 51, pp. 117–122, Jan. 2008.
- [89] G. E. Hinton and R. R. Salakhutdinov, “Reducing the dimensionality of data with neural networks,” *Science*, vol. 313, pp. 504–507, July 2006.
- [90] Y. H. Habiboglu, O. Gunay, and A. Cetin, “Covariance matrix-based fire and flame detection method in video,” *Machine Vision and Applications*, vol. 23, no. 6, pp. 1103–1113, 2012.
- [91] T. Celik, “Fast and efficient method for fire detection using image processing,” *ETRI Journal*, vol. 32, pp. 881–890, December 2010.
- [92] T.-H. Chen, P.-H. Wu, and Y.-C. Chiou, “An early fire-detection method based on image processing,” in *Proc. of International Conference on Image Processing (ICIP)*, vol. 3, pp. 1707–1710, October 2004.

- [93] I. Phillips, W., M. Shah, and N. Da Vitoria Lobo, “Flame recognition in video,” in *Proc. of Fifth IEEE Workshop on Applications of Computer Vision*, pp. 224–229, 2000.
- [94] J. C. Owruksy, D. A. Steinhurst, C. P. Minor, S. L. Rose-Pehrsson, F. W. Williams, and D. T. Gottuk, “Long wavelength video detection of fire in ship compartments,” *Fire Safety Journal*, vol. 41, no. 4, pp. 315–320, 2006.
- [95] B. Töreyin, R. Cinbiş, Y. Dedeoğlu, and A. Çetin, “Fire detection in infrared video using wavelet analysis,” *Optical Engineering*, vol. 46, no. 6, pp. 067204–067204, 2007.
- [96] S. Verstockett, I. Kypraios, P. De Potter, C. Poppe, and R. Van de Walle, “Wavelet-based multi-modal fire detection,” in *Proc. of European Signal Processing Conference (EUSIPCO)*, September 2011.
- [97] B. Kulis and K. Grauman, “Kernelized locality-sensitive hashing for scalable image search,” in *Proc. of IEEE International Conference on Computer Vision (ICCV)*, pp. 2130–2137, Sept 2009.
- [98] C.-C. Chang and C.-J. Lin, *LIBSVM: a library for support vector machines*, 2001. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.
- [99] N. McFarlane and C. Schofield, “Segmentation and tracking of piglets in images,” *Machine Vision and Applications*, vol. 8, pp. 187–193, 1995.
- [100] E. Monari and T. Pollok, “A real-time image-to-panorama registration approach for background subtraction using pan-tilt-cameras,” in *Proc. of International Conference on Advanced Video and Signal-Based Surveillance (AVSS)*, pp. 237–242, Aug 2011.
- [101] L. Zeng, S. Zhang, J. Zhang, and Y. Zhang, “Dynamic image mosaic via sift and dynamic programming,” *Machine Vision and Applications*, vol. 25, no. 5, pp. 1271–1282, 2014.

- [102] M. Uyttendaele, A. Eden, and R. Szeliski, “Eliminating ghosting and exposure artifacts in image mosaics,” in *Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. II, (Kauai, Hawaii), pp. 509–516, IEEE Computer Society, December 2001.
- [103] A. Bevilacqua and P. Azzari, “High-Quality Real Time Motion Detection Using PTZ Cameras,” in *Proc. of IEEE International Conference on Video and Signal Based Surveillance (AVSS)*, pp. 23–23, Nov 2006.
- [104] K. Pulli, M. Tico, and Y. Xiong, “Mobile panoramic imaging system,” in *Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 108–115, June 2010.
- [105] M. Fadaeieslam, M. Fathy, and M. Soryani, “Key frames selection into panoramic mosaics,” in *Proc. of International Conference on Information, Communications and Signal Processing (ICICS)*, pp. 1–5, Dec 2009.
- [106] C. Arth, M. Klopschitz, G. Reitmayr, and D. Schmalstieg, “Real-time self-localization from panoramic images on mobile devices,” in *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pp. 37–46, Oct 2011.
- [107] X. Mei, M. Ramachandran, and S. Zhou, “Video background retrieval using mosaic images,” in *Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 2, pp. 441–444, March 2005.
- [108] P. Azzari, L. Di Stefano, and A. Bevilacqua, “An effective real-time mosaicing algorithm apt to detect motion through background subtraction using a PTZ camera,” in *Proc. of IEEE Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pp. 511–516, Sept 2005.
- [109] D. Wagner, A. Mulloni, T. Langlotz, and D. Schmalstieg, “Real-time panoramic mapping and tracking on mobile phones,” in *Proc. of Virtual Reality Conference (VR)*, pp. 211–218, March 2010.
- [110] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, “Speeded-Up Robust Features (SURF),” *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346 – 359, 2008.

- [111] R. Szeliski, “Image alignment and stitching: A tutorial,” Tech. Rep. MSR-TR-2004-92, Microsoft Research, October 2004.
- [112] R. Collins, A. Lipton, T. Kanade, H. Fujiyoshi, D. Duggins, Y. Tsin, D. Tolliver, N. Enomoto, and O. Hasegawa, “A system for video surveillance and monitoring,” Tech. Rep. CMU-RI-TR-00-12, Robotics Institute, Pittsburgh, PA, May 2000.
- [113] B. Zitov and J. Flusser, “Image registration methods: a survey,” *Image and Vision Computing*, vol. 21, pp. 977–1000, 2003.
- [114] M. Irani and P. Anandan, “Robust multi-sensor image alignment,” in *Proc. of Sixth International Conference on Computer Vision*, pp. 959–966, jan 1998.
- [115] F. Maes, A. Collignon, D. V, G. Marchal, and P. Suetens, “Multimodality Image Registration by Maximization of Mutual Information,” *IEEE transactions on Medical Imaging*, vol. 16, pp. 187–198, 1997.
- [116] J. H. Lee, Y. S. Kim, D. Lee, D.-G. Kang, and J. B. Ra, “Robust CCD and IR Image Registration Using Gradient-Based Statistical Information,” *IEEE Signal Processing Letters*, vol. 17, pp. 347–350, april 2010.
- [117] A. Dame and E. Marchand, “Second-Order Optimization of Mutual Information for Real-Time Image Registration,” *IEEE Transactions on Image Processing*, vol. 21, no. 9, pp. 4190–4203, 2012.
- [118] E. Haber and J. Modersitzki, “Intensity gradient based registration and fusion of multi-modal images,” in *Methods of Information in Medicine*, pp. 726–733, 2006.
- [119] O. Pauly, N. Padoy, H. Poppert, L. Esposito, and N. Navab, “Wavelet energy map: A robust support for multi-modal registration of medical images,” in *Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2184–2191, 2009.

- [120] J. P. W. Pluim, J. Maintz, and M. Viergever, “Image registration by maximization of combined mutual information and gradient information,” *IEEE Transactions on Medical Imaging*, vol. 19, pp. 809–814, Aug 2000.
- [121] M. J. D. Powell, “An efficient method for finding the minimum of a function of several variables without calculating derivatives,” *The Computer Journal*, vol. 7, no. 2, pp. 155–162, 1964.
- [122] P. Torr and A. Zisserman, “MLE-SAC: A New Robust Estimator with Application to Estimating Image Geometry,” *Computer Vision and Image Understanding*, vol. 78, no. 1, pp. 138 – 156, 2000.
- [123] H. Tagare and M. Rao, “Why does mutual-information work for image registration? a deterministic explanation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PP, no. 99, pp. 1–1, 2014.
- [124] J. E. Fowler, “The Redundant Discrete Wavelet Transform and Additive Noise,” *IEEE Signal Processing Letters*, vol. 12, no. 9, pp. 629–632, 2005.
- [125] I. Selesnick, R. Baraniuk, and N. Kingsbury, “The dual-tree complex wavelet transform,” *IEEE Signal Processing Magazine*, vol. 22, pp. 123–151, NOV 2005.
- [126] L. Zhang, D. Zhang, X. Mou, and D. Zhang, “Fsim: A feature similarity index for image quality assessment,” *IEEE Transactions on Image Processing*, vol. 20, pp. 2378–2386, Aug 2011.
- [127] A. Cetin, A. Bozkurt, O. Gunay, Y. Habiboglu, K. Kose, I. Onaran, M. Tofighi, and R. Sevimli, “Projections onto convex sets (POCS) based optimization by lifting,” in *Proc. of Global Conference on Signal and Information Processing (GlobalSIP)*, pp. 623–623, Dec 2013.
- [128] H. Bauschke and J. Borwein, “On Projection Algorithms for Solving Convex Feasibility Problems,” *SIAM Review*, vol. 38, no. 3, pp. 367–426, 1996.
- [129] P. Combettes, “The foundations of set theoretic estimation,” *Proceedings of the IEEE*, vol. 81, pp. 182–208, Feb 1993.

- [130] S. Klein, M. Staring, and J. P. W. Pluim, “Comparison of gradient approximation techniques for optimisation of mutual information in nonrigid registration,” in *Proceedings of SPIE*, vol. 5747, pp. 192–203, 2005.
- [131] B. S. Reddy and B. N. Chatterji, “An FFT-based technique for translation, rotation, and scale-invariant image registration,” *IEEE Transactions on Image Processing*, vol. 5, no. 8, pp. 1266–1271, 1996.
- [132] K. Deep, K. P. Singh, M. L. Kansal, and C. Mohan, “A real coded genetic algorithm for solving integer and mixed integer optimization problems,” *Applied Mathematics and Computation*, vol. 212, no. 2, pp. 505–518, 2009.
- [133] H. J. Johnson, M. McCormick, L. Ibáñez, and T. I. S. Consortium, *The ITK Software Guide*. Kitware, Inc., third ed., 2013. *In press* <http://www.itk.org/ItkSoftwareGuide.pdf>.
- [134] M. A. Fischler and R. C. Bolles, “Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography,” *Communications of the ACM*, vol. 24, pp. 381–395, June 1981.
- [135] D. C. Youla and H. Webb, “Image restoration by the method of convex projections, Part I-Theory,” *IEEE Transactions on Medical Imaging*, vol. MI-I-2, pp. 81–94, 1982.
- [136] K. Kose, O. Gunay, and A. E. Cetin, “Compressive sensing using the modified entropy functional,” *Digital Signal Processing*, vol. 24, pp. 63–70, Jan. 2014.
- [137] A. Cetin, O. Gerek, and Y. Yardimci, “Equiripple FIR filter design by the FFT algorithm,” *IEEE Signal Processing Magazine*, vol. 14, pp. 60–64, Mar 1997.
- [138] C. Pang, “Set intersection problems: supporting hyperplanes and quadratic programming,” *Mathematical Programming*, vol. 149, no. 1-2, pp. 329–359, 2015.

- [139] J. A. Snyman, *Practical mathematical optimization : an introduction to basic optimization theory and classical and new gradient-based algorithms*. Applied optimization, New York: Springer, 2005.
- [140] L. Armijo, “Minimization of functions having lipschitz continuous first partial derivatives,” *Pacific Journal of Mathematics*, vol. 16, no. 1, pp. 1–3, 1966.
- [141] R. Fletcher, *Practical Methods of Optimization; (2Nd Ed.)*. New York, NY, USA: Wiley-Interscience, 1987.
- [142] J. F. Bonnans, Gilbert, C. Lemaréchal, and C. Sagastizábal, *Numerical Optimization – Theoretical and Practical Aspects*. Universitext, Springer Verlag, Berlin, 2006.
- [143] J. Lee, X. Cai, C. Schönlieb, and D. Coomes, “Non-parametric image registration of airborne lidar, hyperspectral and photographic imagery of forests,” *CoRR*, vol. abs/1410.0226, 2014.
- [144] A. Beck and M. Teboulle, “A fast iterative shrinkage-thresholding algorithm for linear inverse problems,” *SIAM Journal on Imaging Sciences*, vol. 2, pp. 183–202, Mar. 2009.
- [145] MATLAB, *Image Processing Toolbox User’s Guide*. 24 Prime Park Way, Natick, MA 01760-1500, 2014.

Appendix A

Selection of Distance Function for Mutual Information

We propose to use distance functions between the marginal distributions and the uniform distribution instead of the entropy function. When Euclidean distance is used the proposed measure is similar to $\mathcal{L}_p$ deviation function. In Figure A.1 different distance functions are compared to the entropy function. for a two-parameter probability distribution. The distances are Euclidean, Cosine, L1-norm, Minkowski (L1-norm to the power of $P = 3$), Chebychev (supremum of absolute difference between the vectors), Bregman divergence with using the entropy function, Renyi entropy with parameter $\alpha = 0.5$, Renyi entropy with parameter $\alpha = 2$.

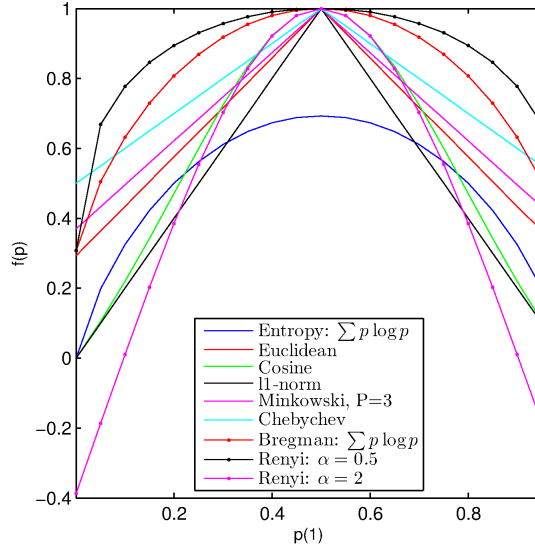


Figure A.1: Distance functions.

The distance functions are compared in a multi-modal image registration experiment. Five different image sets are used and the optimization method is a quasi-Newton method. Table A.1 shows error rates obtained using each distance function. Table A.2 shows optimization times obtained using each distance function. From the results, we see that although the results are comparable l1-norm method has the best and consistent performance.

Table A.1: Optimization errors (sum of the absolute differences between the actual parameter values and optimization results) for different cost functions on various image sets.

	Image Set 1	Image Set 2	Image Set 3	Image Set 4	Image Set 5
Entropy	0.0059	0.0702	0.3179	0.0072	0.0898
Euclidean	0.0027	0.0762	0.6950	0.0006	1.1308
Cosine	13.7166	13.3417	0.7410	3.4502	15.3530
l1-norm	0.0041	0.0494	0.4288	0.0007	0.0867
Minkowski, P=3	0.0016	0.0476	1.0572	3.2613	0.0896
Chebychev	17.4773	9.9197	1.0177	0.0052	7.4826
Bregman	0.0059	0.0687	0.6153	0.0072	0.7224
Renyi, $\alpha = 0.5$	5.5110	13.3352	0.1687	0.0056	0.0990
Renyi, $\alpha = 0.5$	0.0101	0.0549	0.6611	0.0060	0.0993

Table A.2: Optimization times (sec) for different cost functions on various image sets.

	Image Set 1	Image Set 2	Image Set 3	Image Set 4	Image Set 5
Entropy	20.2180	18.3718	34.8942	19.7568	23.9115
Euclidean	30.4174	23.5020	21.9296	23.6257	27.3870
Cosine	21.8899	8.7419	33.6875	17.0526	26.6441
l1-norm	21.7948	20.9126	20.3821	27.1487	20.7408
Minkowski, P=3	27.3367	40.7803	30.1473	22.0825	32.5101
Chebyshev	19.4784	22.5110	34.0945	21.8694	20.2501
Bregman	20.1320	27.4031	23.2158	21.3800	28.9041
Renyi, $\alpha = 0.5$	30.7552	9.0875	27.5367	20.2177	32.5562
Renyi, $\alpha = 0.5$	19.6066	17.6317	28.8828	18.7907	23.4543

Appendix B

Convex Optimization Algorithms based on Projections onto Convex Sets

Projections onto convex sets (POCS) is a widespread method with various areas of application ranging from image registration [135] to beam-forming [14] and compressed sensing [136]. The main idea of POCS is the fact that successive projections onto convex sets should converge to a vector in the intersection of convex sets [34, 129, 137]. In [127] projections onto the supporting hyperplanes of a cost function are proposed to solve a convex optimization problem when closed form solutions are not available. The concept of projections onto supporting hyperplanes is also used in [138] to accelerate the POCS method.

We describe and improve the POCS based convex function optimization methods that we used in Chapter 6. We propose two new optimization methods. The first method finds the minima of a convex function by performing alternating projections between the convex function and a base hyperplane that is parallel to the origin and is lower than the minimum value of the convex function. The second method performs alternating projections between the convex function and its negative version.

The most important novelty of the proposed projection based methods is the fact that they do not require an update parameter as opposed to standard gradient descent algorithms [139]. It is also possible to include relaxation parameters

to the optimization process but it is not really necessary in all the image and video processing examples that we tried. There are adaptive descent algorithms that automatically change the update parameter at each iteration according to the values of the function and its derivative. Line-search algorithms perform multiple gradient descent updates until a convergence criterion is achieved [140]. There are other algorithms that use combination of the values of the function from previous iterations to increase convergence speed. Fast Iterative Shrinkage Thresholding Algorithm (FISTA), which was originally designed to solve inverse problems works using this principle.

Gradient descent based algorithms are first order methods that only use the gradient information for optimization. Newton’s optimization methods on the other hand use second order information which corresponds to the Hessian matrix in the update iterations [141]. By using the curvature information Newton method can reach convergence much faster. When the Hessian of the function is not readily available, it can be estimated iteratively during iterations. The methods using this idea are called quasi-Newton methods. There are many different quasi-Newton methods that mainly differ in the estimation of the Hessian matrix. The most widely used method is Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm [142].

Proposed methods are compared with gradient descent and its adaptive variants in various optimization tasks. The tasks include, polynomial function minimization, logistic regression optimization for classification and mutual information cost function optimization for remote sensing image registration. The proposed methods show better performance in image registration experiments than the quasi-Newton and genetic algorithm methods. There are many methods in the literature that deal with the registration of remote sensing images [4–6, 143]. These methods either use feature based techniques or standard local/global optimization methods. We propose a fast and robust local optimization method to estimate sub-pixel shift parameters between remote sensing images that can perform better than global optimization methods.

In [5] mutual information is used to register TerraSAR-X and Ikonos satellite images. They use mutual information to determine the rough shift parameters between georeferenced images. Although the method works well, its computational

cost is very high. In [6], a mutual information based image registration method is developed to register multispectral images obtained with the MTI (multispectral thermal imager) satellite. They develop a frequency domain model for mutual information and define a correlation length parameter that limits the initial shift values. In [143] the application of non-parametric image registration methods including mutual information and normalized gradient fields are investigated in alignment of LIDAR, hyperspectral and optical images of forests. In [4] a feature based method is proposed for registration of hyperspectral images.

B.1 Optimization Algorithm

We improve the method introduced in [127] to minimize convex cost functions. This method can be used to find the local minimum of convex functions by applying alternating projections onto the supporting hyperplanes. We investigate two new methods for optimization.

B.1.1 Method 1: Projections onto Base Hyperplane

In this method, alternating projections between a convex function and a base hyperplane are performed to find the minimum of the function. If the projection operator onto the cost function has an analytical solution this minimization method can be implemented as shown in Figure B.1(a), which depicts the one dimensional optimization method. The dimension of the parameter set is increased (lifted) by one and the algorithm starts at the initial point:

$$\mathbf{y}_0 = [\mathbf{x}_0; f_0], \quad (\text{B.1})$$

where last element (f_0) added to the parameter set will carry the value of the cost function during projections. The initial point is first projected onto the “Base Hyperplane” and then onto the cost function. The method is derived from the widely used “projection onto the convex sets” [128, 129] algorithm. Alternating projections onto two convex sets (the cost function f and the “Base Hyperplane”), converge to their intersection or the closest vectors in each set. Projections oscillate between the minimum of the convex function and the base hyperplane, as shown in Figure B.1(a).

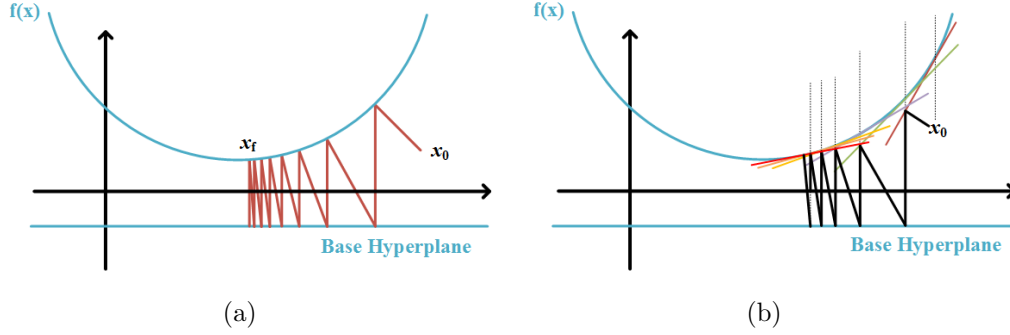


Figure B.1: a) Minimization of a convex function by projections when closed form expression of the projection operator is available, b) minimization of a convex function by projections onto supporting hyperplanes of the cost function.

When the closed form solution to the projection onto the convex function is not available projections onto the supporting hyperplanes can be used. This method of minimization is displayed in Figure B.1(b). First a line is drawn from the initial point of minimization is displayed in Figure B.1(b). This line intersects the cost function at $f_0 = f(x_0)$. Then a supporting hyperplane is defined which is formulated by:

$$\mathbf{a}^T \mathbf{b} = c, \quad (\text{B.2})$$

which is defined at this point with parameters:

$$\mathbf{a} = [\nabla f(\mathbf{x}_0); -1], \quad (\text{B.3})$$

and

$$c = \mathbf{a}^T [\mathbf{x}_0; f_0]. \quad (\text{B.4})$$

If the gradient of the cost function is not available sub-gradients can also be used. After the hyperplane is determined the orthogonal projection of the vector on the base hyperplane, $\mathbf{y}_1 = [\mathbf{x}_0; f_{bh}]$, onto the supporting hyperplane is calculated.

$$\mathbf{y}_2 = \mathbf{y}_1 - \frac{\mathbf{a}(\mathbf{y}_1^T \mathbf{a} - c)}{\|\mathbf{a}\|^2}. \quad (\text{B.5})$$

Then, another line is drawn from the projected point onto the cost function and the algorithm iterates by performing alternating projections between the function and the base hyperplane. The final point, $\mathbf{x}_f$, should be close to the minimum of the cost function.

the supporting hyperplanes as shown in Figure B.3(b). We use the correction technique from the previous section to deal with non-decreasing function issues.

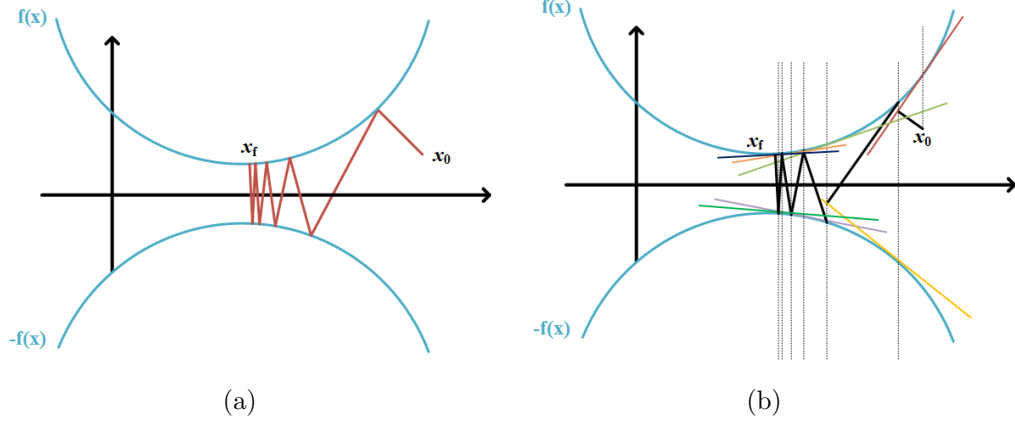


Figure B.3: a) Minimization of a convex function by projections when closed form expression of the projection operator is available, b) minimization of a convex function by projections onto supporting hyperplanes of the cost function.

B.2 Experimental Results

The proposed methods are compared with three different algorithms in a well-established image processing problem. The algorithms are gradient descent with constant step size, steepest descent with Armijo step size rule [140] and FISTA method with line-search algorithm [144]. Gradient descent is the standard method used in the minimization of convex functions:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \mu \nabla f(\mathbf{x}_k), \quad (\text{B.8})$$

where μ is the constant update parameter. The determination of this parameter is usually done by trial and error. The projection based optimization algorithm does not need an update parameter that needs to be tuned. Typically, the steepest descent method starts with parameters, $\mu = 1$, $\sigma = 0.1$, and $\beta = 0.5$ and decreases the update parameter with $\mu = \beta\mu$ until the following condition is satisfied:

$$\frac{f(\mathbf{x}_k - \mu \nabla f(\mathbf{x}_k)) - f(\mathbf{x}_k)}{\mu} < -\sigma \nabla f(\mathbf{x}_k)^T \nabla f(\mathbf{x}_k). \quad (\text{B.9})$$

This search is performed at each iteration and this requires more computational power.

FISTA algorithm [144] with constant update parameter, L , starts with $t_1 = 1$ and $\mathbf{z}_1 = \mathbf{x}_0$ and updates the samples as follows:

$$\mathbf{x}_k = \mathbf{z}_k - \frac{1}{L} \nabla f(\mathbf{z}_k), \quad (\text{B.10})$$

$$t_{k+1} = \frac{1 + \sqrt{1 + 4t_k^2}}{2}, \quad (\text{B.11})$$

$$\mathbf{z}_{k+1} = \mathbf{x}_k + \frac{t_k - 1}{t_{k+1}} (\mathbf{x}_k - \mathbf{x}_{k-1}). \quad (\text{B.12})$$

A line-search method can be incorporated into FISTA to adaptively change the update parameter. Line-search method starts with $\eta > 0$ and sets $L = \eta L$ until the following condition is satisfied:

$$f(\mathbf{x}_k) < f(\mathbf{z}_k) + (\mathbf{x}_k - \mathbf{z}_k)^T \nabla f(\mathbf{z}_k) + \frac{L}{2} \|\mathbf{x}_k - \mathbf{z}_k\|^2. \quad (\text{B.13})$$

B.2.1 Polynomial Function Minimization Experiments

We compare five different algorithms in polynomial function minimization problems. POCS1 stands for the first proposed method and POCS2 stands for the second proposed method. STEEP stands for the steepest descent method with Armijo step size rule. GD stands for standard gradient descent method. and FISTA stands for the fast iterative shrinkage algorithm with line-search update. We use three different functions for optimization. The maximum number of iterations is set to 1000, the gradient norm tolerance is set to 1e-10 and maximum number of correction iterations is set to 40 for all algorithms. Table B.1 shows the iteration count (N), function evaluation count (Nf), gradient evaluation count (Ng), the running time of the algorithms and the norm of the parameter error. From the results we see that all algorithms have comparable performance and proposed methods surpass other algorithms in some cases.

Table B.1: Optimization results for polynomial functions. Times are in seconds.

$f_1(\mathbf{x}) = \gamma_1(\mathbf{x}(1) - \psi_1)^4 + \kappa$					
	N	Nf	Ng	Error	Time
POCS1	1000	1001	1001	0.0043	0.1446
POCS2	1000	2001	1001	0.0045	0.2102
STEEP	1000	1007	1001	0.0100	0.1253
GD	1000	0001	1002	0.1000	0.0993
FISTA	1000	2009	1009	0.0036	0.1963
$f_2(\mathbf{x}) = \gamma_1(\mathbf{x}(1) - \psi_1)^2 + \gamma_2(\mathbf{x}(2) - \psi_2)^4 + (\mathbf{x}(1) - \psi_1)(\mathbf{x}(2) - \psi_2) + \kappa$					
	N	Nf	Ng	Error	Time
POCS1	1000	3360	1001	0.2282	0.2392
POCS2	1000	4831	0503	0.2282	0.3209
STEEP	1000	40639	1001	0.2282	1.5567
GD	1000	0001	1002	0.2988	0.1352
FISTA	1000	2013	1013	0.2288	0.2766
$f_3(\mathbf{x}) = \gamma_1(\mathbf{x}(1) - \psi_1)^2 + \gamma_2(\mathbf{x}(2) - \psi_2)^4 + \gamma_3(\mathbf{x}(3) - \psi_3)^6 + \kappa$					
	N	Nf	Ng	Error	Time
POCS1	1000	2755	1001	0.0766	0.2311
POCS2	1000	3610	0520	0.0809	0.2203
STEEP	1000	1016	1001	0.0804	0.1581
GD	1000	0001	1002	0.9022	0.1223
FISTA	1000	2018	1018	0.1391	0.2422

Figure B.4 shows the value of the function versus iterations for $f_3(\mathbf{x})$. We observe that steepest descent has the fastest initial convergence but POCS based methods achieve better results eventually. Since FISTA is not a descent method, it oscillates for this problem.

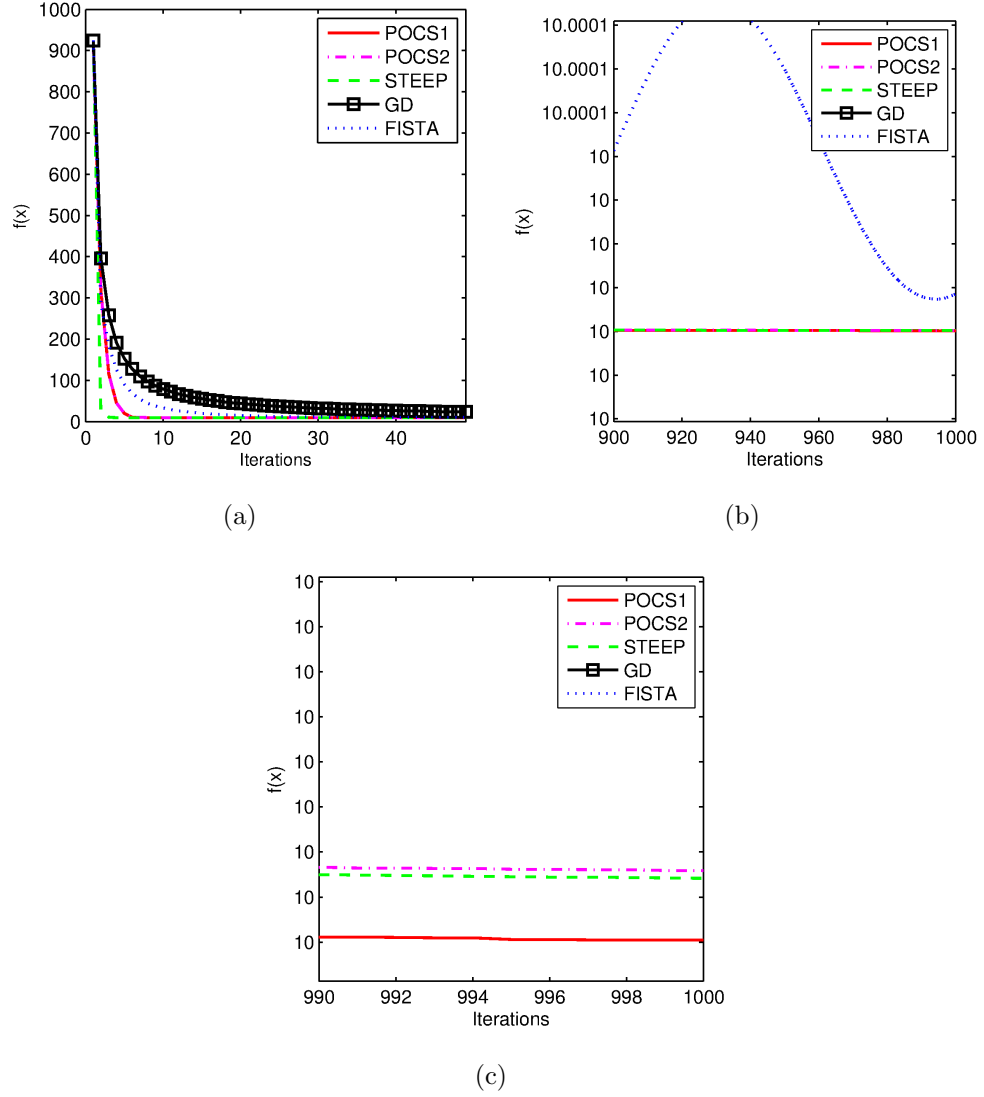


Figure B.4: Function value versus iterations for $f_3(\mathbf{x})$, a) iterations 0-50, b) iterations, 950-1000, c) iterations 990-1000.

B.2.2 Logistic Regression Experiments

We use logistic regression hypothesis for separable classification problems. For this problem, we have a number of training pairs, $\{\mathbf{x}_i, y_i\}$ where $\mathbf{x}$ are the samples and y denotes the class of the sample. Logistic function can be defined as follows:

$$h_{\mathbf{w}}(\mathbf{x}) = \frac{1}{1 + e^{-\mathbf{w}^T \mathbf{x}}}, \quad (\text{B.14})$$

where $\mathbf{w}$ denotes the coefficients of the separating hyperplane. The following convex cost function can be used in gradient descent applications:

$$C(h_{\mathbf{w}}(\mathbf{x}), y) = \begin{cases} -\log(h_{\mathbf{w}}(\mathbf{x})) & \text{if } y = 1 \\ -(1 - \log(h_{\mathbf{w}}(\mathbf{x}))) & \text{if } y = 0 \end{cases} \quad (\text{B.15})$$

The cost function corresponding to the training set with M samples is:

$$J(\mathbf{w}) = \frac{1}{M} \sum_{i=1}^M C(h_{\mathbf{w}}(\mathbf{x}_i), y_i). \quad (\text{B.16})$$

We test the training and classification performance of the algorithms with standard datasets from the UCI (University of California, Irvine) machine learning repository [71] and libsvm test datasets [98]. We use half of the samples for training and the rest for testing. The maximum number of iterations is set to 1000, the gradient norm tolerance is set to 1e-3 and maximum number of correction iterations is set to 40 for all algorithms.

Table B.2 gives descriptions of datasets (Nsamples: number of samples, Nfeatures: number of features) and training optimization results. The error is defined as the ratio of misclassified samples in the test set. We observe that all methods have very close error rates for all datasets. The proposed methods have comparable performance in terms of the number of iterations and computation time. Figure B.5 depicts the value of the cost function versus iterations for diabetes_scale dataset.

Table B.2: Training optimization and classification results for logistic regression experiments. Times are in seconds.

Dataset:australian_scale					
Nsamples: 690 Nfeatures:14					
	N	Nf	Ng	Error	Time
POCS1	0890	6477	0891	0.0400	0.7247
POCS2	0894	7437	0449	0.0400	0.7553
STEEP	0889	6244	0890	0.0400	0.7370
GD	1000	0001	1002	0.0380	0.2713

FISTA	0892	1797	0905	0.0400	0.5044
Dataset:diabetes_scale					
Nsamples: 768 Nfeatures:8					
POCS1	0199	1349	0200	0.0940	0.1474
POCS2	0235	1863	0119	0.0940	0.1825
STEEP	0263	1804	0264	0.0940	0.2017
GD	1000	0001	1002	0.0940	0.2489
FISTA	0580	1173	0593	0.0940	0.3047
Dataset:fourclass_scale					
Nsamples: 862 Nfeatures:2					
POCS1	0018	0091	0019	0.1320	0.0100
POCS2	0022	0140	0013	0.1320	0.0133
STEEP	0015	0075	0016	0.1320	0.0083
GD	1000	0001	1002	0.1320	0.2034
FISTA	0025	0059	0034	0.1320	0.0122
Dataset:german.numer_scale					
Nsamples: 1000 Nfeatures:24					
POCS1	0487	4141	0488	0.1070	0.6430
POCS2	0485	4607	0244	0.1070	0.6552
STEEP	0598	5472	0599	0.1070	0.8514
GD	1000	0001	1002	0.1100	0.4375
FISTA	0839	1694	0855	0.1070	0.7829
Dataset:mushrooms					
Nsamples: 3916 Nfeatures:112					
POCS1	1000	1001	1001	0.0000	1.6126
POCS2	1000	2001	1001	0.0000	1.9228
STEEP	0330	0341	0331	0.0000	0.5240
GD	1000	0001	1002	0.0000	1.5515
FISTA	1000	2019	1019	0.0000	3.5896
Dataset:ionosphere_scale					
Nsamples: 351 Nfeatures:34					

POCS1	1000	5611	1001	0.0310	0.6410
POCS2	1000	6590	0502	0.0310	0.6431
STEEP	1000	5888	1001	0.0310	0.6770
GD	1000	0001	1002	0.0360	0.2546
FISTA	1000	2013	1013	0.0300	0.5193
Dataset:sonar_scale					
Nsamples: 208 Nfeatures:60					
POCS1	1000	3470	1001	0.0270	0.4660
POCS2	1000	4541	0512	0.0260	0.4805
STEEP	1000	5904	1001	0.0270	0.6612
GD	1000	0001	1002	0.0300	0.2585
FISTA	1000	2013	1013	0.0270	0.5499
Dataset:a2a					
Nsamples: 2265 Nfeatures:119					
POCS1	1000	7958	1001	0.2290	1.8163
POCS2	1000	9282	0502	0.2290	1.8237
STEEP	1000	9538	1001	0.2260	2.1537
GD	1000	0001	1002	0.2310	0.6213
FISTA	1000	2017	1017	0.2300	1.3767

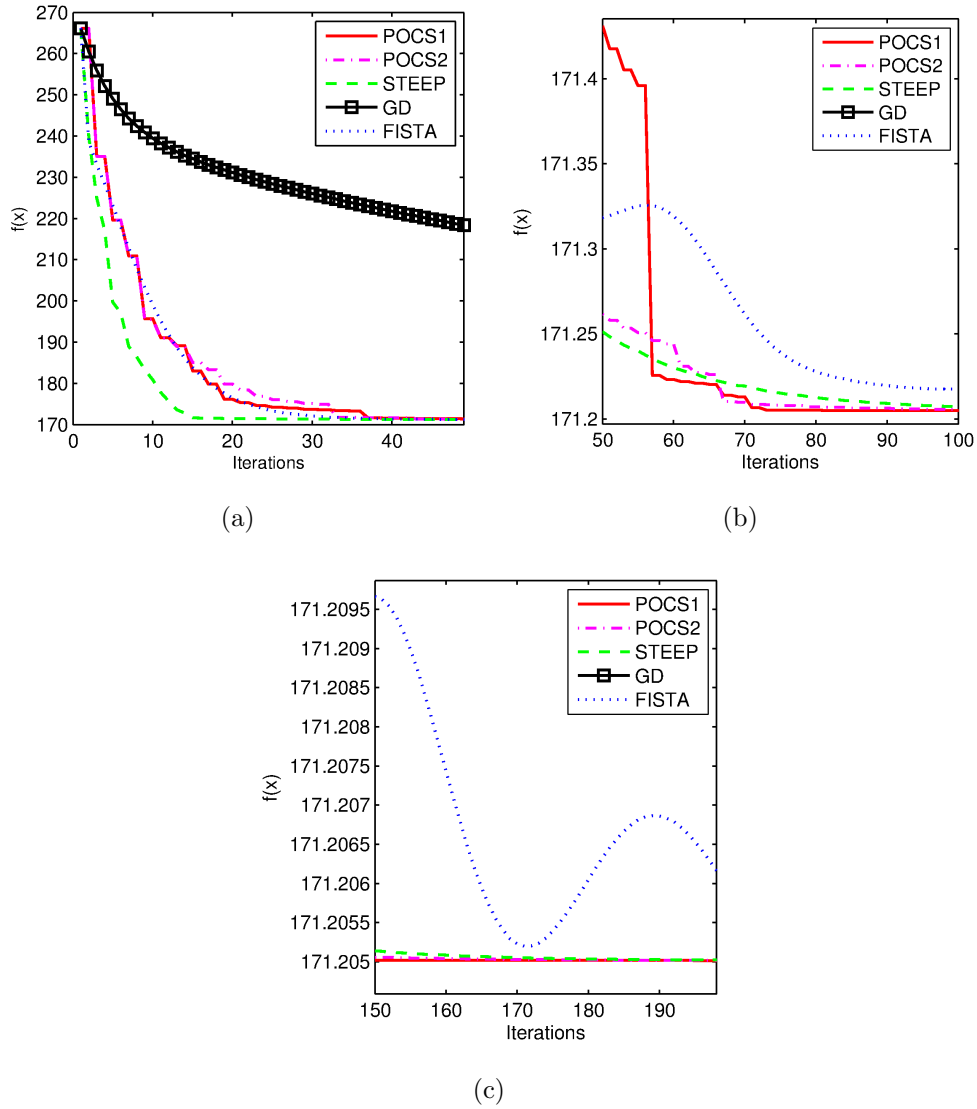


Figure B.5: Function value versus iterations for diabetes_scale dataset, a) iterations 0-50, b) iterations, 50-100, c) iterations 150-200.

B.2.3 Image Registration Experiments

We use a mutual information cost function for registration of multi-modal remote sensing images. In most remote sensing application the images are geometrically aligned, but there might still be georeferencing translation errors [5]. The mutual information cost function is not necessarily convex, it can be defined as quasi-convex for some of the parameters [22]. We use numerical techniques to estimate

the gradient of mutual information, as described in [130].

We also compare the proposed algorithms to quasi-Newton and genetic algorithm (GA) based methods. Newton methods can be described as follows:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \lambda[\mathbf{H}_k]^{-1}\nabla f(\mathbf{x}_k), \quad (\text{B.17})$$

where $\mathbf{H}_k$ denotes the Hessian matrix of the function. When the Hessian matrix is not available it can be approximated iteratively. BFGS method performs the updates as follows:

$$\begin{aligned} &\text{Solve for } \mathbf{d}_k : \\ &\mathbf{H}_k \mathbf{d}_k = \nabla f(\mathbf{x}_k) \\ &\text{Use line-search to find } \mu_k : \\ &\mathbf{x}_{k+1} = \mathbf{x}_k - \mu_k \nabla f(\mathbf{x}_k) \\ &\mathbf{m}_k = \mu_k \mathbf{d}_k \\ &\mathbf{q}_k = \nabla f(\mathbf{x}_{k+1}) - \nabla f(\mathbf{x}_k) \\ &\mathbf{H}_{k+1} = \mathbf{H}_k + \frac{\mathbf{q}_k \mathbf{q}_k^T}{\mathbf{q}_k^T \mathbf{m}_k} - \frac{\mathbf{H}_k \mathbf{m}_k \mathbf{m}_k^T \mathbf{H}_k}{\mathbf{m}_k^T \mathbf{H}_k \mathbf{m}_k} \end{aligned} \quad (\text{B.18})$$

Genetic algorithms (GA) are global optimization methods used with non-smooth cost functions. For image registration problem we use a mixed-integer genetic algorithm which is faster for quantized variables [132].

For this experiment the maximum number of iterations is set to 50, the gradient norm tolerance is set to 10^{-3} and maximum number of correction iterations is set to 40 for all algorithms. Error is defined as the norm parameter-wise error. Table B.3 shows the results of image registration experiments. In the table QN denotes the results of the quasi-Newton BFGS method. We use the Matlab implementation of the method in the function “*fminunc*” [145]. We set the tolerances for the QN method to 10^{-20} .

In the experiments, we use four hyperspectral Landsat images [4]. These are pre-registered and we manually translate one of the images to test the optimization methods. We use an image set from MTI (Multispectral Thermal Imager) satellite [6]. We also use an image set obtained from TerraSAR-X and Ikonos satellites [5]. These image do not have ground truth values, therefore we manually choose control points to estimate ground truth values to calculate the error rates of the algorithms.

We see that for this experiment proposed POCS based methods perform much better than competing algorithms. Except for two cases it also performs better than the GA method. The error rate of other algorithms is too large and as can be seen in Figure B.6, they do not usually converge to a correct solution. Although the POCS based algorithms are local optimization methods, they are usually faster and more successful then GA based global optimization method for this problem. Figure B.7 shows the Landsat images of Iowa [4] used for the last image set. Figure B.8 shows the TerraSAR-X and Ikonos images. Figure B.9 shows two band of MTI images.

Table B.3: Multi-modal Image registration results.
Times are in seconds.

Landsat hyper-spectral images [4]					
	N	Nf	Ng	Error	Time
POCS1	0050	0440	0051	0.1855	31.2227
POCS2	0050	0467	0019	0.1855	27.5677
STEEP	0050	0815	0051	20.9387	55.8685
GD	0050	0001	0052	21.1546	12.0945
FISTA	0050	0173	0123	21.1535	35.7891
QN	0005	0054	0054	20.9396	7.8577
GA	0000	0421	0000	0.3066	54.1843
Landsat hyper-spectral images [4]					
POCS1	0050	0300	0051	0.9955	6.3308
POCS2	0050	0339	0022	0.9955	5.5421
STEEP	0050	0737	0051	27.9699	14.2257
GD	0050	0001	0052	28.1564	3.2707
FISTA	0050	0175	0125	28.1579	11.8543
QN	0003	0045	0045	27.8727	2.2832
GA	0000	0421	0000	55.9890	17.7102
Landsat hyper-spectral images [4]					
POCS1	0050	0289	0051	1.4041	19.5374
POCS2	0050	0327	0023	1.4041	17.3819

STEEP	0050	0845	0051	19.8512	47.0745
GD	0050	0001	0052	19.7997	10.1146
FISTA	0050	0165	0115	19.7994	34.0117
QN	0003	0024	0024	19.9435	3.6815
GA	0000	0421	0000	73.4002	52.2025
Landsat hyper-spectral images [4]					
POCS1	0050	0172	0051	0.9000	13.7965
POCS2	0050	0247	0033	0.8690	14.3973
STEEP	0050	0508	0051	40.0531	31.0263
GD	0050	0001	0052	40.6163	9.6631
FISTA	0050	0169	0119	40.6061	33.6491
QN	0001	0020	0020	40.6057	3.0984
GA	0000	0421	0000	0.8309	51.2167
MTI (Multispectral Thermal Imager) images [6]					
POCS1	0050	0245	0051	0.2937	5.9313
POCS2	0050	0294	0027	0.2937	5.5890
STEEP	0050	0800	0051	15.3922	15.9099
GD	0050	0001	0052	15.4456	3.4612
FISTA	0050	0173	0123	15.4457	11.9404
QN	0002	0045	0045	15.5219	2.3637
GA	0000	0421	0000	0.4143	18.1546
TerraSAR-X and Ikonos images [5]					
POCS1	0050	0174	0051	0.3517	54.4634
POCS2	0050	0361	0026	1.0615	72.7376
STEEP	0050	0730	0051	12.3181	148.6364
GD	0050	0001	0052	12.4898	34.9767
FISTA	0050	0167	0117	12.3304	118.6974
QN	0002	0045	0045	12.3538	23.7547
GA	0000	0421	0000	0.0491	187.9396

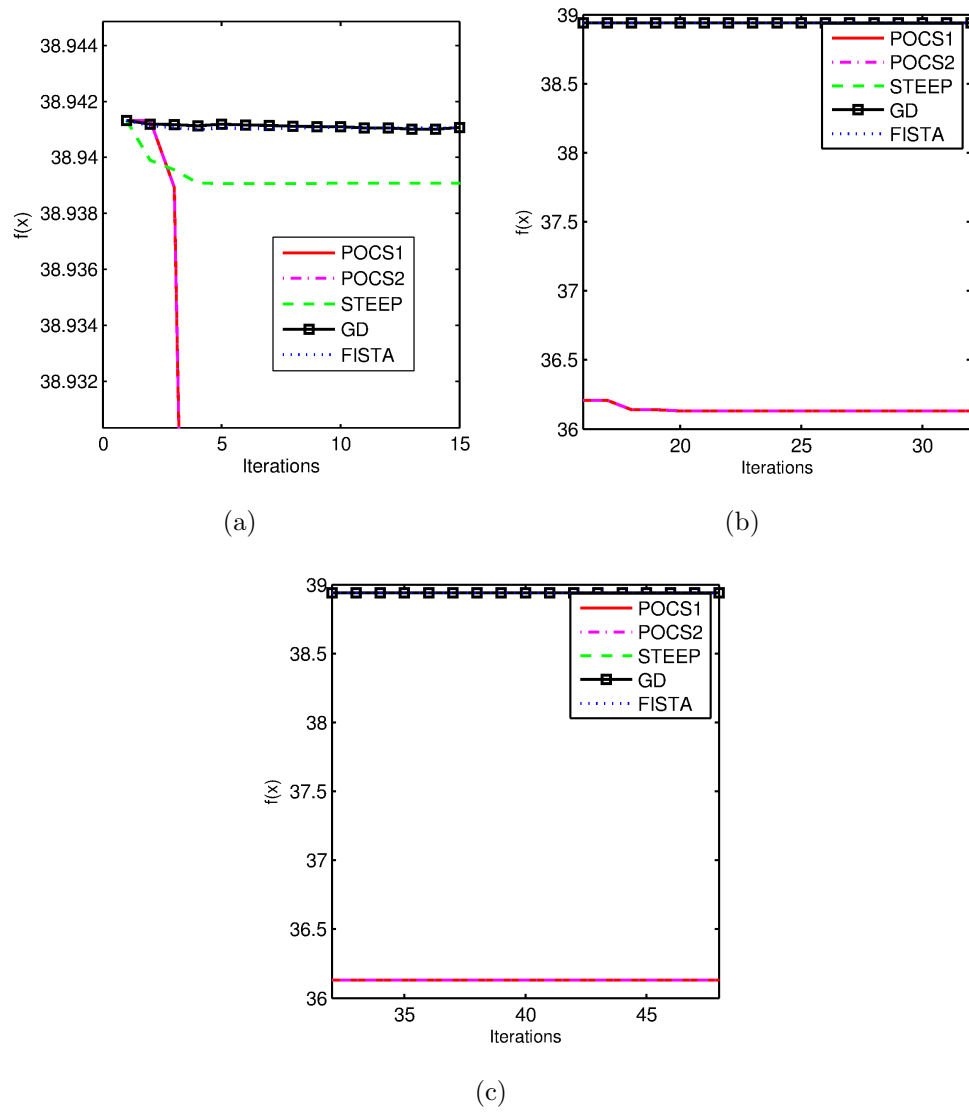


Figure B.6: Function value versus iterations for third image set, a) iterations 0-16, b) iterations, 17-32, c) iterations 33-50.

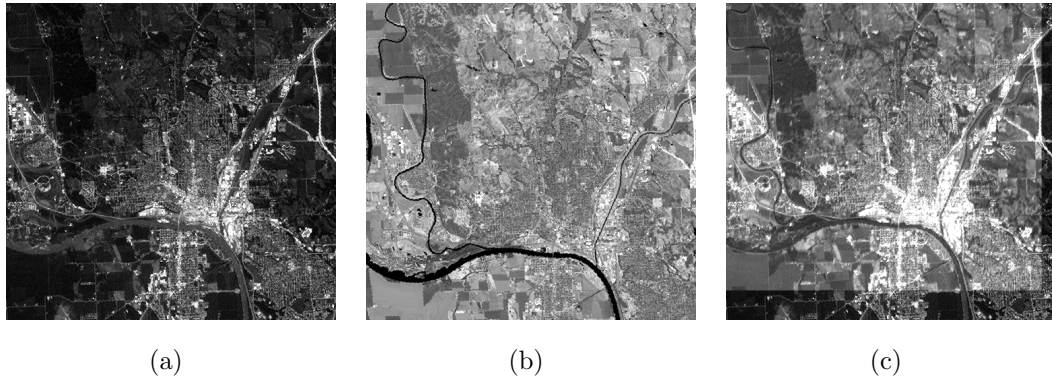


Figure B.7: Example image registration result for Iowa Landsat images [4], a) band 1, b) band 5, c) registered band 5 image.

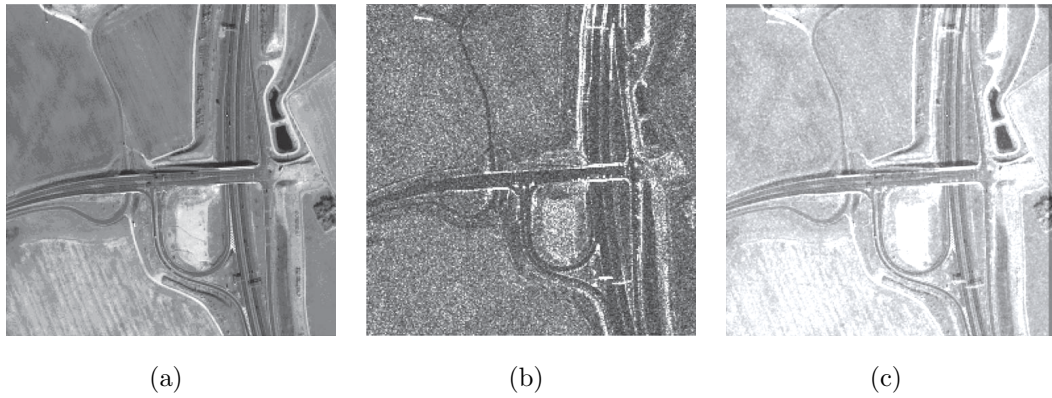


Figure B.8: Example image registration result for TerraSAR-X and Ikonos images [5], a) Ikonos image, b) TerraSAR-X image, c) registered TerraSAR-X image.

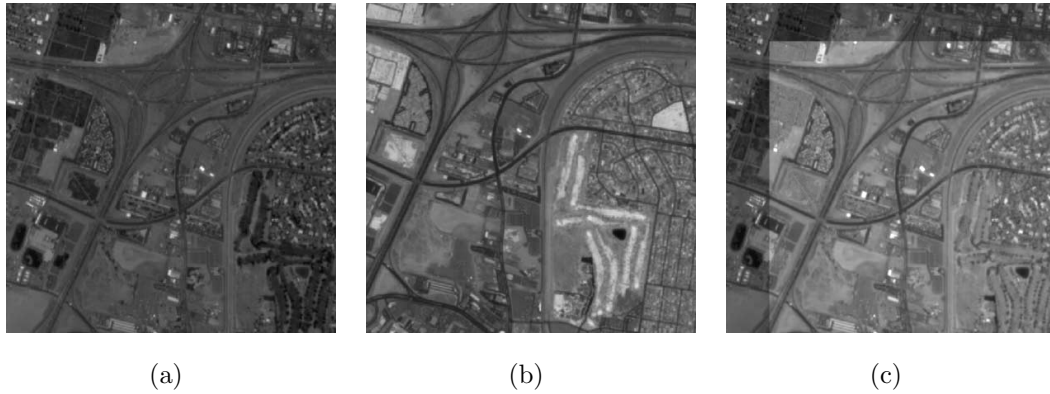
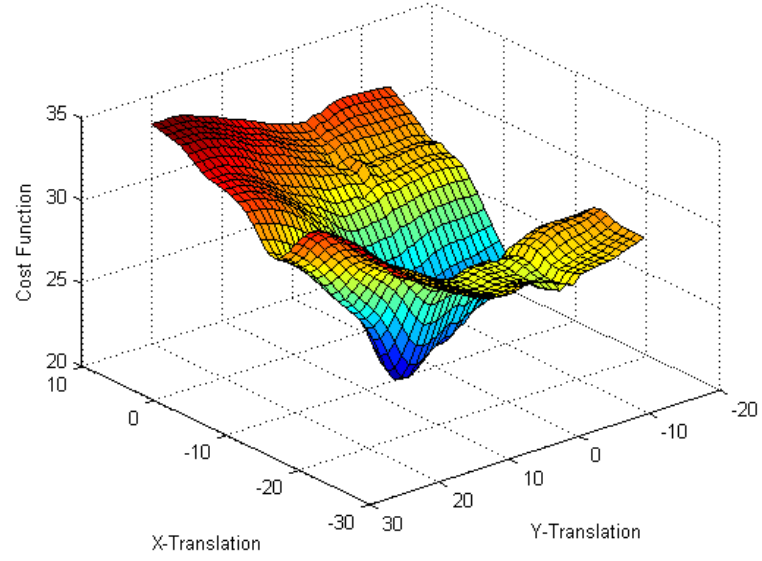
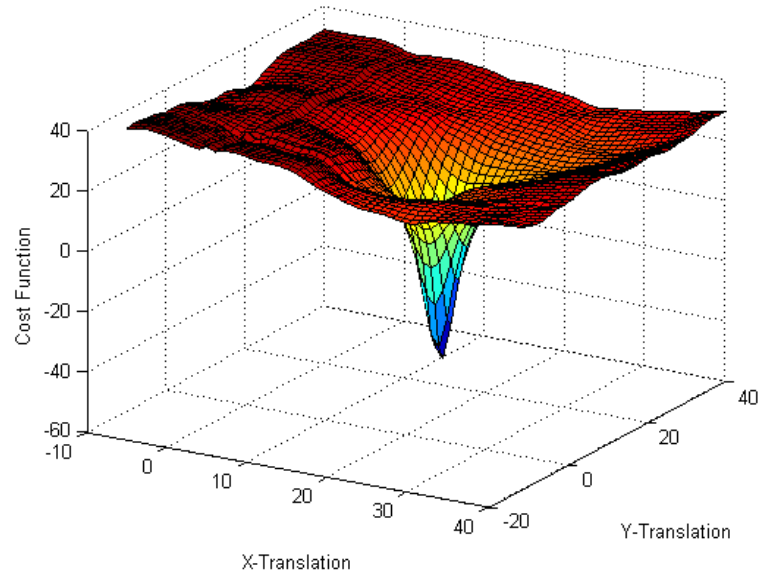


Figure B.9: Example image registration result for MTI images [6], a) band A image, b) band D image, c) registered band D image.

The success of the proposed methods in image registration experiments can be explained by examining the cost function. As seen in Figure B.10 the cost function is not totally smooth and has subtle local minima. While some optimization methods get stuck on local minima the proposed methods can step over by applying projections onto the supporting hyperplanes. Other algorithms can reach the solution only when the initial solution is near the true solution.



(a)



(b)

Figure B.10: Mutual information cost function versus transformation parameters, a) TerraSAR-X/Ikonos images [5], b) MTI images [6].

Figure B.11 shows the evolution of the cost function value versus iterations for

quasi-Newton (*fminunc*) and POCS1 methods for different initial shift parameters. For this experiment the last Landsat image pair is used from Table B.3 and the true shift parameters are $(-33.2, -34.1)$. From the figure, we see that POCS1 method always reaches the minimum of the function regardless of the initial shift parameters. However, quasi-Newton method finds the minima only when the initial parameters are close to true parameters. For other cases the QN method cannot decrease the cost function using its line-search algorithm. We perform the same experiment on the third Landsat image set from Table B.3 with the true shift parameters are $(-10.6, -23.9)$ (see Figure B.12). For this image set, QN method fails for all conditions, POCS1 method fails for only one initial condition.

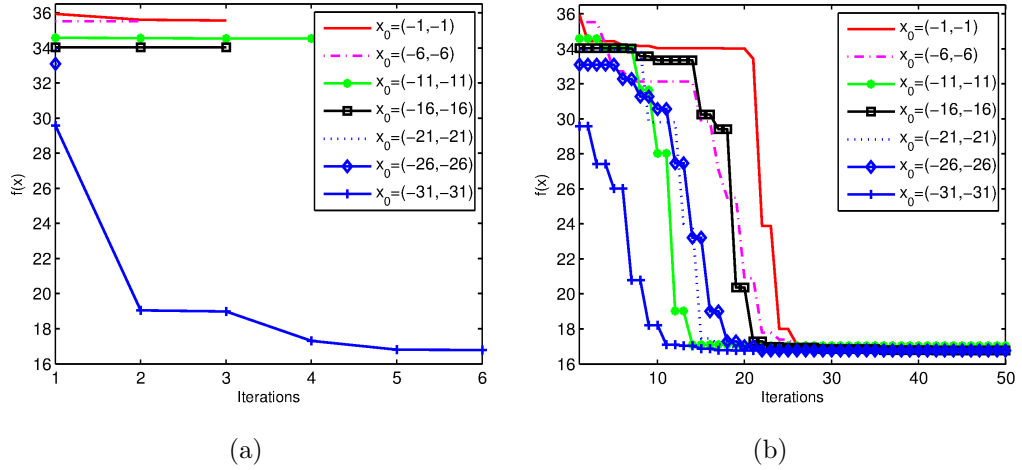
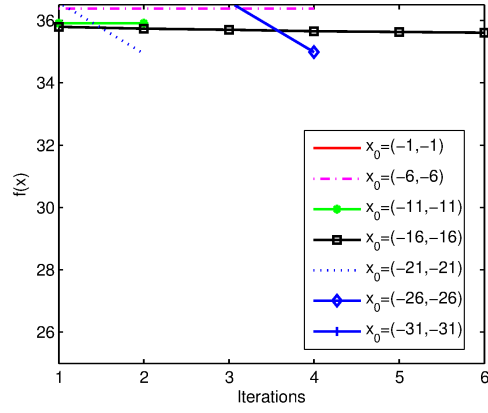
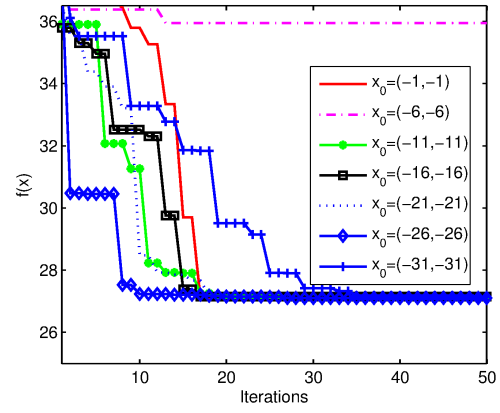


Figure B.11: Mutual information cost function versus iteration for different initial shift parameters on the fourth Landsat image set: a) quasi-Newton method, b) POCS1 method.



(a)



(b)

Figure B.12: Mutual information cost function versus iteration for different initial shift parameters on the third Landsat image set: a) quasi-Newton method, b) POCS1 method.