



A VISION-BASED QUALITY INSPECTION SYSTEM FOR FABRIC DEFECT DETECTION AND CLASSIFICATION

Philibert Nsengiyumva

Thesis submitted in fulfilment of the requirements for the degree

**DOCTOR TECHNOLOGIAE:
ELECTRICAL ENGINEERING**

in the

Department of Electrical, Electronic and Computer Engineering

at the

Central University of Technology, Free State

Promoter: Prof. H.J. Vermaak, PhD

Co-promoter: Dr. N.J. Luwes, DTech

BLOEMFONTEIN
November 2014


To the wonderful family that the Almighty has bestowed upon me

- *My wife Marianne Mukabisangwa*
- *My daughter Naila Uwera Morin*
- *My sons Serge Uwimana, Patrick Uwonkunda and Pacifique Uwizeye*
- *My adoptive daughter Sonia Ineza*

Declaration of independent work

I, **PHILIBERT NSENGIYUMVA**, passport number [REDACTED] and student number [REDACTED], do hereby declare that this research project, being submitted to the Central University of Technology for the degree **DOCTOR TECHNOLOGIAE: ENGINEERING: ELECTRICAL**, is my own independent work; and complies with the Code of Academic Integrity, as well as other relevant policies, procedures, rules and regulations of the Central University of Technology, Free State; and has not been submitted before to any institution by myself or any other person in fulfilment (or partial fulfilment) of the requirements for the attainment of any qualification.

Signature - Philibert Nsengiyumva



SIGNATURE OF STUDENT

5 December 2014
DATE

Acknowledgments

My thanks go first to the Almighty God for all He has done for me throughout my life. Nothing is possible without His intervention.

The Government of Rwanda through the Rwanda Education Board provided me with a loan without which the studies in South Africa would have been impossible.

Kigali Institute of Science and Technology and Central University of Technology, Free State through their partnership facilitated my admission and integration into this doctorate programme. Kigali Institute of Science and Technology, my current employer, provided me with a study leave which allowed me to be absent from work and concentrate on my research. I wish to acknowledge all that important support here.

My deep gratitude goes to my supervisors, Prof. Herman Vermaak and Dr. Nicolaas Luwes. Their guidance, advice and encouragement throughout the whole research process were invaluable. They also provided me with favourable working conditions. Prof. Vermaak, please find here the expression of my gratitude for your understanding and support in my problems, both academic and personal.

Many thanks to my friends and colleagues at the CUT Exchange House for the assistance and social climate they provided to me, which allowed me to progress smoothly in my research.

Last but not least, I wish to thank my family members for enduring my long period of absence.

Abstract

Quality inspection of textile products is an important issue for fabric manufacturers. It is desirable to produce the highest quality goods in the shortest amount of time possible. Fabric faults or defects are responsible for nearly 85% of the defects found by the garment industry. Manufacturers recover only 45 to 65% of their profits from second or off-quality goods. There is a need for reliable automated woven fabric inspection methods in the textile industry.

Numerous methods have been proposed for detecting defects in textile. The methods are generally grouped into three main categories according to the techniques they use for texture feature extraction, namely statistical approaches, spectral approaches and model-based approaches.

In this thesis, we study one method from each category and propose their combinations in order to get improved fabric defect detection and classification accuracy. The three chosen methods are the grey level co-occurrence matrix (GLCM) from the statistical category, the wavelet transform from the spectral category and the Markov random field (MRF) from the model-based category. We identify the most effective texture features for each of those methods and for different fabric types in order to combine them.

Using GLCM, we identify the optimal number of features, the optimal quantisation level of the original image and the optimal intersample distance to use. We identify the optimal GLCM features for different types of fabrics and for three different classifiers.

Using the wavelet transform, we compare the defect detection and classification performance of features derived from the undecimated discrete wavelet and those derived from the dual-tree complex wavelet transform. We identify the best features for different types of fabrics.

Using the Markov random field, we study the performance for fabric defect detection and classification of features derived from different models of Gaussian Markov random fields of order from 1 through 9. For each fabric type we identify the best model order.

Finally, we propose three combination schemes of the best features identified from the three methods and study their fabric detection and classification performance. They lead generally to improved performance as compared to the individual methods, but two of them need further improvement.

Table of contents

Declaration of independent work.....	ii
Acknowledgments.....	iii
Abstract.....	iv
Table of contents.....	v
List of acronyms and abbreviations	xi
List of figures.....	xiii
List of tables.....	xix
Chapter 1: Introduction.....	1
1.1 Importance of textile fabrics in human life.....	1
1.2 Fabrics and the process of fabrication	1
1.3 Importance of fabric defect detection and classification	3
1.4 The fabric defects and their classes	3
1.5 Commercial automatic defect detection systems.....	7
1.6 Research in the field of fabric defect detection and classification	8
1.7 Outstanding problems (challenges) of fabric defect detection and classification	11
1.8 The objective of this thesis	12
1.9 The contribution of this thesis	13
1.10 The outline of this thesis.....	13
Chapter 2: Review of fabric defect detection literature	15
2.1 Introduction.....	15
2.2 Methods based on first order statistics.....	15
2.3 Methods based on second order statistics	16
2.3.1 Grey level co-occurrence matrices.....	17
2.3.2 Autocorrelation function	20
2.3.3 Grey level run length method.....	20
2.4 Spectral methods.....	21
2.4.1 Methods based on the Fourier transform	21
2.4.2 Methods based on Gabor filters	22
2.4.3 Methods based on wavelet transforms	24
2.4.4 Methods based on the dual-tree complex wavelet transform.....	28
2.5 Model-based methods	29

2.6	Summary.....	31
Chapter 3: Review of the methods employed in this thesis		33
3.1	Introduction.....	33
3.2	Grey level co-occurrence matrix features	33
3.2.1	Definition of grey level co-occurrence matrix	33
3.2.2	Compilation of a co-occurrence feature of a grey level image	34
3.2.3	Co-occurrence matrix features	35
3.2.4	Methods to improve the GLCM-related computational cost	38
3.3	Wavelet transform and wavelet transform features	40
3.3.1	Introduction to wavelet transform.....	40
3.3.2	Discrete wavelet transform	41
3.3.3	Multiresolution approximations	42
3.3.4	Wavelet representations	44
3.3.5	FWT algorithm.....	45
3.3.6	The undecimated discrete wavelet transform.....	46
3.3.7	Two-dimensional wavelet transform	47
3.3.8	The dual-tree complex wavelet transform	49
3.3.9	The wavelet transform features	52
3.4	Markov random field model-based features	53
3.4.1	What is a Markov random field?.....	53
3.4.2	Neighbourhood systems	53
3.4.3	Markov random field models	55
3.4.4	MRF texture modelling	57
3.4.5	MRF texture features and feature extraction.....	58
3.5	Classifiers	59
3.5.1	Pattern recognition basic concepts	59
3.5.2	Feature normalisation.....	60
3.5.3	Euclidean distance classifier	61
3.5.4	K-nearest-neighbour classifier	63
3.5.5	Feed-forward neural network classifier	64
3.5.6	Multiclass classifiers	66
3.6	Summary.....	67
Chapter 4: Description of the experimental data		71
4.1	Introduction.....	71
4.2	The fabric classes and defect types.....	72

4.3	Elementary experimental sample.....	75
4.4	Labelling of the samples	76
4.5	Training and testing data sets	77
4.6	Performance evaluation	78
4.7	Summary.....	80
Chapter 5: Exploration of grey level co-occurrence matrix features.....		81
5.1	Introduction.....	81
5.2	Optimal number of the GLCM features for fabric defect detection	81
5.2.1	Introduction.....	81
5.2.2	Description of the experiment.....	83
5.2.3	Results and interpretation.....	86
5.3	Optimal GLCM feature sets per fabric type	88
5.3.1	Introduction.....	88
5.3.2	The algorithm.....	90
5.3.3	Results and their evaluation	92
5.4	Performance of the GLCM features as function of the quantisation level of the image.....	96
5.4.1	Introduction.....	96
5.4.2	Description of the experiment.....	97
5.4.3	Results and interpretation.....	97
5.5	Performance of the GLCM features as function of the interpixel distance ...	99
5.5.1	Introduction.....	99
5.5.2	Description of the experiment.....	100
5.5.3	Results and interpretation.....	100
5.6	Effect of the classifier on the performance of GLCM features	102
5.6.1	Introduction.....	102
5.6.2	Dependence of the optimal feature set on the classifier.....	103
5.6.3	Defect detection performance for different classifiers.....	109
5.6.4	Generalisation ability of classifiers.....	113
5.7	Defect classification using GLCM features.....	114
5.7.1	Introduction.....	114
5.7.2	Description of the experiment.....	114
5.7.3	Results and interpretation.....	116
5.8	Modified GLCHS for fast computation of GLCM features	119

5.9	Summary.....	122
Chapter 6: Exploration of wavelet transform features.....		124
6.1	Introduction.....	124
6.2	Which wavelet transform to use?	124
6.3	Design of the wavelet for the UDWT.....	125
6.4	Design of the dual-tree complex wavelet filters.....	128
6.4.1	Fundamentals of dual-tree complex wavelet design.....	128
6.4.2	Q-shift design solution.....	130
6.5	Which wavelet features to use?	132
6.5.1	Feature sets for UDWT.....	132
6.5.2	Feature sets for DTCWT.....	133
6.5.3	Feature normalisation.....	134
6.6	Feature extraction algorithm.....	134
6.6.1	Feature extraction algorithm for UDWT	135
6.6.2	Feature extraction algorithm for DTCWT	136
6.7	Compared defect detection performance of DTCWT and UDWT.....	137
6.8	Defect detection performance of wavelet features vs the wavelet size of support	140
6.8.1	DTCWT-based features	140
6.8.2	UDWT-based features.....	142
6.9	Defect detection performance of wavelet features vs the number of wavelet decomposition levels	143
6.10	Summary of best wavelet-based features per fabric types.....	145
6.11	Effect of the classifier on the performance of wavelet features	146
6.11.1	Comparison of UDWT vs DTCWT for different fabric types when the KNN and the neural network classifiers are used	147
6.11.2	Dependence of the defect detection rate on the wavelet size of support when the KNN and the neural network classifiers are used	148
6.11.3	Best wavelet filter length for different fabric types when the KNN and the neural network classifiers are used	149
6.11.4	Defect detection performance of wavelet-based features with respect to the number of wavelet decomposition levels when the KNN and the neural network classifiers are used	154
6.11.5	Summarised comparison of the best-based wavelet features per fabric type for the three classifiers	156
6.12	Fabric defect classification using wavelet features	158

6.12.1 Defect classification results of UDWT-based features	159
6.12.2 Defect classification results of DTCWT-based features	162
6.12.3 Summary of best wavelet-based features for defect classification	165
6.13 Summary	167
Chapter 7: Exploration of Markov random field-based features	169
7.1 Introduction	169
7.2 Which Markov random field model?	169
7.3 MRF feature extraction algorithm	169
7.4 Defect detection performance as function of the model order	172
7.5 Effect of the classifier on the performance of MRF features	174
7.6 Fabric defect classification using MRF features	178
7.7 Summary	182
Chapter 8: Combination schemes and their performance	184
8.1 Introduction	184
8.2 Direct combination of the best selected features (CS1)	185
8.2.1 Introduction	185
8.2.2 Experimental steps	186
8.2.3 Results and interpretation	187
8.3 GLCM and MRF features from DTCWT wavelet coefficients (CS2)	189
8.3.1 Description of the method and the experiment	189
8.3.2 Results and interpretation	191
8.4 GLCM and MRF features from DTCWT reconstructed images (CS3)	193
8.4.1 Description of the method and the experiment	193
8.4.2 Results and interpretation	195
8.5 Comparison of the three combination schemes	196
8.6 Effect of different classifiers	198
8.6.1 Combined schemes using the KNN classifier	198
8.6.2 Combined schemes using the feed-forward neural network classifier ...	200
8.7 Defect classification using the combined methods	202
8.8 Summary	206
Chapter 9: Conclusions and future work	208
9.1 Conclusions	208
9.1.1 Conclusions regarding GLCM features	208
9.1.2 Conclusions regarding wavelet features	209



9.1.3 Conclusions regarding MRF features	210
9.1.4 Conclusions regarding combination schemes	211
9.2 Recommendations for future work	212
Bibliography	213

List of acronyms and abbreviations

ACF	Autocorrelation function
ANFIS	Adaptive neural-fuzzy inference system
AVA	All-versus-all
CWT	Continuous wavelet transform
DWT	Discrete wavelet transform
DTCWT	Dual-tree complex wavelet transform
ECOC	Error-correcting output-coding
EFM	Elementary feature matrix
FFT	Fast Fourier transform
FWT	Fast wavelet transform
FPGA	Field programmable gate array
GIS	Golden image subtraction
GLCHH	Grey level co-occurrence hybrid histogram
GLCIA	Grey level co-occurrence integrated algorithm
GLCLL	Grey level co-occurrence linked list
GLCM	Grey level co-occurrence matrix
GLCHS	Grey level co-occurrence hybrid structure
GLDM	Grey level difference method
GLRL	Grey level run length
GRLM	Grey level run length matrix
GMRF	Gaussian Markov random field
KNN	K-nearest-neighbour
LSE	Least squares estimate
NGDM	Neighbouring grey level dependence matrix
MCE	Minimum classification error
MI	Mutual information
ML	Maximum likelihood
MLL	Multilevel logistic
MRF	Markov random field
mRMR	Minimum redundancy maximum relevance
MSWAR	Multiscale wavelet representation
OVA	One-versus-all
PCA	Principal component analysis
SAR	Synthetic aperture radar

TILDA	Textile texture database developed by the Texture Analysis working group of the Deutsche Forschungsgemeinschaft in Germany
UDWT	Undecimated discrete wavelet transform

List of figures

Figure 1-1: Basic components of a woven fabric [2].....	2
Figure 1-2: A simplified sketch of a two-harness loom [1, 2]	2
Figure 1-3: Flowchart of woven fabric defects based on their source [9]	4
Figure 1-4: Block diagram of a fabric inspection system [16]	9
Figure 1-5: Detailed block diagram of a fabric inspection system [16]	10
Figure 3-1: Construction of the co-occurrence matrix P^0 for $d=1$ [116].	34
Figure 3-2: Examples of mother wavelets [124].....	41
Figure 3-3: Single-level DWT decomposition.....	45
Figure 3-4: Single-level DWT decomposition and reconstruction	46
Figure 3-5: Three-level DWT decomposition.....	46
Figure 3-6: Single-level UDWT decomposition and reconstruction	47
Figure 3-7: Two-level UDWT decomposition and reconstruction. H_{0up} , H_{1up} , F_{0up} , F_{1up} are upsampled versions of H_0 , H_1 , F_0 and F_1 respectively	47
Figure 3-8: One level wavelet decomposition of a 2D signal.....	48
Figure 3-9: Wavelet decomposition of an image	48
Figure 3-10: Example of wavelet transform of an image	49
Figure 3-11: DTCWT decomposition of a signal using two real filter bank trees	51
Figure 3-12: Neighbourhood systems up to the fifth order.....	54
Figure 3-13: Cliques for a second order neighbourhood system	54
Figure 3-14: Typical pattern recognition system	60
Figure 3-15: Principle of the Euclidean distance classifier	62
Figure 3-16: MCE training of an Euclidean distance classifier	63
Figure 3-17: Feed-forward neural network with R inputs, M neurons in the hidden layer and N outputs [149].....	64
Figure 4-1: Structure of the dataset used in this thesis	72
Figure 4-2: Representatives of the fabric classes contained in the TILDA dataset	74
Figure 4-3: Division of a fabric image into elementary samples.....	76
Figure 5-1: Mean of best detection rates as function of the number of GLCM features for all the samples in the dataset	86
Figure 5-2: Average of best detection rates as function of the number of GLCM features for the different fabric types	88

Figure 5-3: Computation of the cumulative square error of the detection rates for the C1R1 fabric type and the {f1, f2, f3} triplet of GLCM features.....	91
Figure 5-4: Defect detection rate obtained using the best triplet, the optimal triplet as well as the whole set of fourteen GLCM features for the fabric types C1R1, C1R3, C2R2 and C2R3.....	94
Figure 5-5: Defect detection rate obtained using the best triplet, the optimal triplet as well as the whole set of fourteen GLCM features for the fabric types C3R1, C3R3, C4R1 and C4R4.....	95
Figure 5-6: Detection rate as function of the quantisation level for the samples in the training set and for different types of fabrics	98
Figure 5-7: Detection rate as function of the quantisation level for the samples in the testing set and for different types of fabrics.....	99
Figure 5-8: Detection rate as function of the interpixel distance for the samples in the training set and for different types of fabrics	101
Figure 5-9: Detection rate as function of the interpixel distance for the samples in the testing set and for different types of fabrics.....	102
Figure 5-10: Mean of the best detection rates as function of the number of GLCM features for all the samples in the dataset when the MCE-trained Euclidean distance classifier, K-nearest-neighbour classifier and feed-forward neural network classifier are used	105
Figure 5-11: Mean best detection rates as function of the number of GLCM features for the samples from the different types of fabric when a MCE-trained Euclidean distance classifier is used.	106
Figure 5-12: Mean best detection rates as function of the number of GLCM features for the samples from the different types of fabric when a K-nearest-neighbour classifier is used.	107
Figure 5-13: Mean best detection rates as function of the number of GLCM features for the samples from the different types of fabric when a feed-forward neural network classifier is used.	107
Figure 5-14: Compared detection rate of the optimal feature set to the full set (14 features) when the MCE-trained Euclidean distance classifier is used ..	111
Figure 5-15: Compared detection rate of the optimal feature set to the full set (fourteen features) when the K-nearest-neighbour classifier is used.....	112

Figure 5-16: Compared detection rate of the optimal feature set to the full set (fourteen features) when the feed-forward neural network classifier is used	112
Figure 5-17: Generalisation ability of different classifiers	113
Figure 5-18: Single-level defect classification using GLCM features.....	116
Figure 5-19: Average classification rate (%) for different types of fabrics using different classifiers.....	117
Figure 5-20: Data structures of GLCHS	120
Figure 6-1: A two-channel nonsampled filter bank	126
Figure 6-2: Implementation of 1D dual-tree complex wavelet transform using FIR real coefficient filters	129
Figure 6-3: Comparison of detection rate of DTCWT- and UDWT-based features as the wavelet filter length increases	138
Figure 6-4: Comparison of detection rate of DTCWT- and UDWT-based features for the eight types of fabrics	139
Figure 6-5: Comparison of detection performance of DTCWT- and UDWT-based features for the fabrics of types C4R1 and C4R3 for different wavelet support sizes	140
Figure 6-6: Detection rate of DTCWT-based features as the wavelet filter length increases for the different types of fabric.....	141
Figure 6-7: Detection rate of UDWT-based features as the wavelet filter length increases for the eight different types of fabric.....	142
Figure 6-8: Variation of average defect detection rate with respect to the number of wavelet decomposition levels	144
Figure 6-9: Variation of average defect detection rate with respect to the number of UDWT wavelet decomposition levels for fabric types C4R1 and C4R3	145
Figure 6-10: Comparison of detection rate of DTCWT- and UDWT-based features for the eight types of fabrics using the K-nearest-neighbour classifier	147
Figure 6-11: Comparison of detection rate of DTCWT- and UDWT-based features for the eight types of fabrics using the feed-forward neural network classifier	148
Figure 6-12: Comparison of detection rate of DTCWT- and UDWT-based features as the wavelet filter length increases when a K-nearest-neighbour classifier is used	148

Figure 6-13: Comparison of detection rate of DTCWT- and UDWT-based features as the wavelet filter length increases when a feed-forward neural network classifier is used	149
Figure 6-14: Detection rate of DTCWT-based features as the wavelet filter length increases for the different types of fabric when a KNN classifier is used	150
Figure 6-15: Detection rate of DTCWT-based features as the wavelet filter length increases for the different types of fabric when a feed-forward neural network classifier is used	151
Figure 6-16: Detection rate of UDWT-based features as the wavelet filter length increases for the different types of fabric when a KNN classifier is used	152
Figure 6-17: Detection rate of UDWT-based features as the wavelet filter length increases for the different types of fabric when a feed-forward neural network classifier is used	153
Figure 6-18: Variation of average defect detection rate with respect to the number of wavelet decomposition levels when a K-nearest-neighbour classifier is used	155
Figure 6-19: Variation of average defect detection rate with respect to the number of wavelet decomposition levels when a feed-forward neural network classifier is used	155
Figure 6-20: Comparison of the classifiers about their defect detection capability from wavelet-based features	158
Figure 6-21: Detection classification rate of UDWT-based features as the wavelet filter length increases for the different types of fabric when a MCE-trained Euclidean distance classifier is used	160
Figure 6-22: Detection classification rate of UDWT-based features as the wavelet filter length increases for the different types of fabric when a K-nearest-neighbour classifier is used	161
Figure 6-23: Detection classification rate of UDWT-based features as the wavelet filter length increases for the different types of fabric when a feed-forward neural network classifier is used	162

Figure 6-24: Detection classification rate of DTCWT-based features as the wavelet filter length increases for the different types of fabric when MCE-trained Euclidean distance classifier is used	163
Figure 6-25: Detection classification rate of DTCWT-based features as the wavelet filter length increases for the different types of fabric when a K-nearest-neighbour classifier is used	164
Figure 6-26: Detection classification rate of DTCWT-based features as the wavelet filter length increases for the different types of fabric when a feed-forward neural network classifier is used	165
Figure 6-27: Detection classification rate of DTCWT-based features as the wavelet filter length increases for the different types of fabric when feed-forward neural network classifier is used	166
Figure 7-1: Neighbourhood of point X	170
Figure 7-2: Defect detection performance of MRF features as the order of the model changes for different fabric types.....	173
Figure 7-3: Defect detection performance of MRF features as the order of the model changes for different fabric types when a K-nearest-neighbour classifier is used	175
Figure 7-4: Defect detection performance of MRF features as the order of the model changes for different fabric types when a feed-forward neural network classifier is used	177
Figure 7-5: Defect classification performance of MRF features as the order of the model changes when a MCE-trained Euclidean distance classifier is used	179
Figure 7-6: Defect classification performance of MRF features as the order of the model changes when a K-nearest-neighbour classifier is used.....	180
Figure 7-7: Defect classification performance of MRF features as the order of the model changes when a feed-forward neural network classifier is used..	181
Figure 8-1: Block diagram of the first combination scheme (CS1).....	186
Figure 8-2: Defect detection performance of CS1 features for different fabric types as number of principal components of wavelet features changes	187
Figure 8-3: Average defect detection performance of combined method CS1 compared to the individual methods	189
Figure 8-4: Block diagram of the second combination scheme (CS2)	190

Figure 8-5: Defect detection performance of combined method CS2 for different fabric types as number of principal components of wavelet features changes.....	191
Figure 8-6: Defect detection performance of combined method CS2 compared to the individual methods.....	193
Figure 8-7: Block diagram of the third combination scheme (CS3).....	194
Figure 8-8: Defect detection performance of combined method CS3 compared to the individual methods.....	195
Figure 8-9: Comparison of defect detection performance of the three combined methods	197
Figure 8-10: Defect detection performance of combined methods CS1, CS2 and CS3 compared to the individual GLCM-, wavelet- and MRF-based methods when the K-nearest-neighbour classifier is used.....	199
Figure 8-11: Defect detection performance of combined methods CS1, CS2 and CS3 compared to the individual GLCM-, wavelet- and MRF-based methods when the feed-forward neural network classifier is used.....	202
Figure 8-12: Defect classification performance of combined methods CS1, CS2 and CS3 compared to the individual GLCM-, wavelet- and MRF-based methods when the MCE-trained Euclidean distance classifier is used...	203
Figure 8-13: Defect classification performance of combined methods CS1, CS2 and CS3 compared to the individual GLCM-, wavelet- and MRF-based methods when the K-nearest-neighbour classifier is used	204
Figure 8-14: Defect classification performance of combined methods CS1, CS2 and CS3 compared to the individual GLCM-, wavelet- and MRF-based methods when the feed-forward neural network classifier is used	205

List of tables

Table 1-1: Common fabric defects [9].....	5
Table 4-1: The TILDA images used in this thesis	72
Table 4-2: Fabric classes in the TILDA dataset.....	73
Table 4-3: Fabric defects observed in the TILDA dataset	73
Table 4-4: Algorithm for sample labelling	77
Table 4-5: Number of samples in the experimental dataset.....	78
Table 4-6: Confusion matrix	79
Table 5-1: Groups of samples and their respective number of elements	83
Table 5-2: Feature table of the samples of C1R1E1 group of the training set.....	84
Table 5-3: Algorithm for evaluating the highest detection performance obtained with a single GLCM feature.....	85
Table 5-4: Average of best detection rates as function of the number of GLCM features	87
Table 5-5: Best triplets of GLCM features for defect detection in different types of fabrics.....	89
Table 5-6: Algorithm for identifying the optimal triplet of GLCM features for each fabric type	92
Table 5-7: Optimal GLCM features.....	93
Table 5-8: Detect performance of the optimal triplet of GLCM features with respect to the best GLCM triplet and the whole GLCM feature set.....	96
Table 5-9: Detection rate (%) vs quantisation level for the training set	98
Table 5-10: Detection rate (%) vs quantisation level for the testing set	99
Table 5-11: Detection rate (%) vs interpixel distance for the training set	101
Table 5-12: Detection rate (%) vs interpixel distance for the testing set.....	102
Table 5-13: Mean of the best detection rates as function of the number of GLCM features for different classifiers.....	105
Table 5-14: Number of optimal GLCM features and average of the best detection rates for different classifiers.....	108
Table 5-15: Optimal GLCM feature sets for different classifiers for the eight fabric types	109
Table 5-16: Average defect classification rates (%) obtained using GLCM features	117

Table 5-17: Sample confusion matrices for defect classification using GLCM features	119
Table 6-1: Decomposition filters h0 and h1 for UDWT wavelets.....	128
Table 6-2: Filters h0 for different complex wavelets.....	131
Table 6-3: Algorithm for feature extraction using UDWT	136
Table 6-4: Average detection rate (%) of DTCWT- and UDWT-based features as the wavelet filter length increases	138
Table 6-5: Average detection rate (%) of DTCWT- and UDWT-based features for the eight fabric types.....	140
Table 6-6: Average detection rate (%) of DTCWT-based features as the wavelet filter length increases for the eight types of fabrics.....	142
Table 6-7: Average detection rate (%) UDWT-based features as the wavelet filter length increases for the 8 types of fabrics.....	143
Table 6-8: Best wavelet-based features per fabric type	146
Table 6-9: Average detection rate (%) of DTCWT-based features as the wavelet filter length increases for the eight types of fabrics when a K-nearest-neighbour classifier is used	150
Table 6-10: Average detection rate (%) of DTCWT-based features as the wavelet filter length increases for the eight types of fabrics when a feed-forward neural network classifier is used	151
Table 6-11: Average detection rate (%) of UDWT-based features as the wavelet filter length increases for the eight types of fabrics when a K-nearest-neighbour classifier is used	152
Table 6-12: Average detection rate (%) of UDWT-based features as the wavelet filter length increases for the eight types of fabrics when the feed-forward neural network classifier is used	153
Table 6-13: Best wavelet-filter length when the three different classifiers are used.	154
Table 6-14: Best wavelet filter features for different fabrics using the three classifiers	156
Table 6-15: Topology of the neural network classifier used for wavelet-based defect classification.....	159
Table 6-16: Average defect classification rate (%) of UDWT-based features as the wavelet filter length increases for the eight types of fabrics when a MCE-trained Euclidean distance classifier is used	160

Table 6-17: Average defect classification rate (%) of UDWT-based features as the wavelet filter length increases for the eight types of fabrics when a K-nearest-neighbour classifier is used	161
Table 6-18: Average defect classification rate (%) of UDWT-based features as the wavelet filter length increases for the eight types of fabrics when a feed-forward neural network classifier is used	162
Table 6-19: Average defect classification rate (%) of DTCWT-based features as the wavelet filter length increases for the eight types of fabrics when MCE-trained Euclidean distance classifier is used	163
Table 6-20: Average defect classification rate (%) of DTCWT-based features as the wavelet filter length increases for the eight types of fabrics when a K-nearest-neighbour classifier is used	164
Table 6-21: Average defect classification rate (%) of DTCWT-based features as the wavelet filter length increases for the eight types of fabrics when a feed-forward neural network classifier is used	165
Table 6-22: Comparison of best wavelet-based defect classification performance of three different classifiers	166
Table 6-23: Average defect classification rate (%) of DTCWT-based features as the wavelet filter length increases for the eight types of fabrics when a feed-forward neural network classifier is used	167
Table 7-1: Number of GMRF model parameters.....	171
Table 7-2: Algorithm for GMRF feature extraction from a given image f and for a given model order	172
Table 7-3: Average defect detection rate (%) for different fabric as the order of the MRF model used to find texture features varies from 1 through 9.....	174
Table 7-4: Best MRF-based features per fabric type	174
Table 7-5: Average defect detection rate (%) for different fabric as the order of the MRF model used to find texture features varies from 1 through 9 and when a K-nearest-neighbour classifier is used.....	176
Table 7-6: Average defect detection rate (%) for different fabric as the order of the MRF model used to find texture features varies from 1 through 9 and when a feed-forward neural network is used	177
Table 7-7: Comparison of best MRF-based defect detection performance of three different classifiers	178

Table 7-8: Average defect classification rate (%) for different fabric as the order of the MRF model changes when a MCE-trained Euclidean distance classifier is used	179
Table 7-9: Average defect classification rate (%) for different fabric as the order of the MRF model changes when a K-nearest-neighbour classifier is used	180
Table 7-10: Average defect classification rate (%) for different fabric as the order of the MRF model changes when a feed-forward neural network classifier is used	181
Table 7-11: Comparison of best MRF-based defect classification performance of three different classifiers	182
Table 8-1: Most effective features for the fabric defect detection methods to be combined when a MCE-trained Euclidean distance classifier is used....	185
Table 8-2: Average defect detection rate (%) of CS1 features for different fabric types as number of principal components of wavelet features changes	188
Table 8-3: Average defect detection rate (%) of combined method CS1 compared to the individual GLCM-, wavelet- and MRF-based methods.....	189
Table 8-4: Average defect detection rate (%) of combined method CS2 for different fabric types as number of principal components of GLCM and MRF changes	192
Table 8-5: Average defect detection rate (%) of combined method CS2 compared to the individual GLCM-, wavelet- and MRF-based methods.....	193
Table 8-6: Average defect detection rate (%) of combined method CS3 compared to the individual GLCM-, wavelet- and MRF-based methods.....	196
Table 8-7: Compared average defect detection rates (%) of the three combined methods	197
Table 8-8: Recommended defect detection combined method for each fabric type..	198
Table 8-9: Most effective features for the fabric defect detection methods to be combined when K-nearest-neighbour classifier is used.....	199
Table 8-10: Average defect detection rate (%) of combined methods CS1, CS2 and CS3 compared to the individual GLCM-, wavelet- and MRF-based methods when the K-nearest-neighbour classifier	200
Table 8-11: Most effective features for fabric defect detection methods to be combined when the feed-forward neural classifier is used	201

Table 8-12: Average defect detection rate (%) of combined methods CS1, CS2 and CS3 compared to the individual GLCM-, wavelet- and MRF-based methods when the feed-forward neural classifier is used	202
Table 8-13: Average defect classification rate (%) of combined methods CS1, CS2 and CS3 compared to the individual GLCM-, wavelet- and MRF-based methods when the MCE-trained Euclidean distance classifier is used...	203
Table 8-14: Confusion matrices comparing defect classification for fabric type C1R1 using GLCM features and CS1 combined features using the MCE-trained Euclidean distance classifier (There are less misclassified samples i.e. off-diagonal elements in the confusion matrix (b))	204
Table 8-15: Average defect classification rate (%) of combined methods CS1, CS2 and CS3 compared to the individual GLCM-, wavelet- and MRF-based methods when K-nearest-neighbour classifier is used.....	205
Table 8-16: Average defect classification rate (%) of combined methods CS1, CS2 and CS3 compared to the individual GLCM-, wavelet- and MRF-based methods when feed-forward neural network classifier is used	206

Chapter 1: Introduction

1.1 Importance of textile fabrics in human life

Textile fabrics are found in every aspect of human life, from the basic needs of people to the most advanced technological invention.

The basic needs of people are food, clothing and shelter. For example, in the food industry fabrics are used to provide plant covers, absorbent liners in pre-packaged meats and reusable cloth bags [1]. The most common use of fabrics is in clothing. They are used to provide warmth, cover, protection and aesthetic properties. Fabrics are also used as shelter in form of tents and umbrellas and are part of building materials.

Other uses of fabrics include carpets, conveyer belts in factories, fabric-supported hoses at petrol stations, space suits for astronauts etc. It is difficult to imagine any sector of human life that does not make use of textile fabrics.

1.2 Fabrics and the process of fabrication

According to Kadolph, a fabric is a pliable, plane-like structure that can be made into two- or three-dimensional products that require some shaping and flexibility [2]. Most fabrics are made from yarns and are either woven or knitted. Yarns themselves are continuous threadlike strands composed of fibres that have been twisted together during a process known as spinning. Fibres are the smallest part of the fabric; they are fine, hair-like substances, categorised as either natural or manufactured [1]. Examples of natural fibres are cotton, wool and silk. Manufactured fibres are formed from chemicals and comprise acrylic, nylon and polyester.

Woven fabrics are made with two or more sets of yarns interlaced at right angles. Figure 1-1 shows the two basic components of a woven fabric; warp yarns and filling yarns. The yarns in the length-wise direction are called warp yarns or ends while the yarns in the cross-wise direction are called filling yarns, weft or picks.

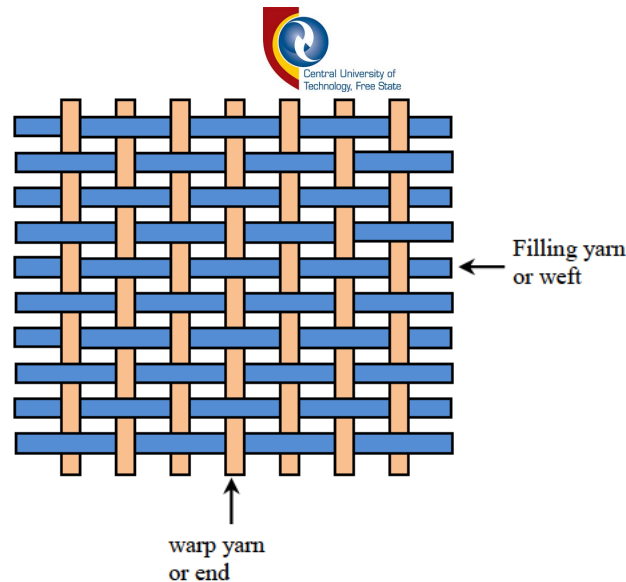


Figure 1-1: Basic components of a woven fabric [2]

Woven fabrics are produced on a loom. Figure 1-2 shows a simplified sketch of a two-harness loom. The process of producing woven fabric is as follows [1]. The warp yarns are first set in place from the warp beam through the harnesses to the cloth roll. The harness is a frame that holds many thin vertical wires called heddles, each with a hole in the middle through which passes a warp yarn.

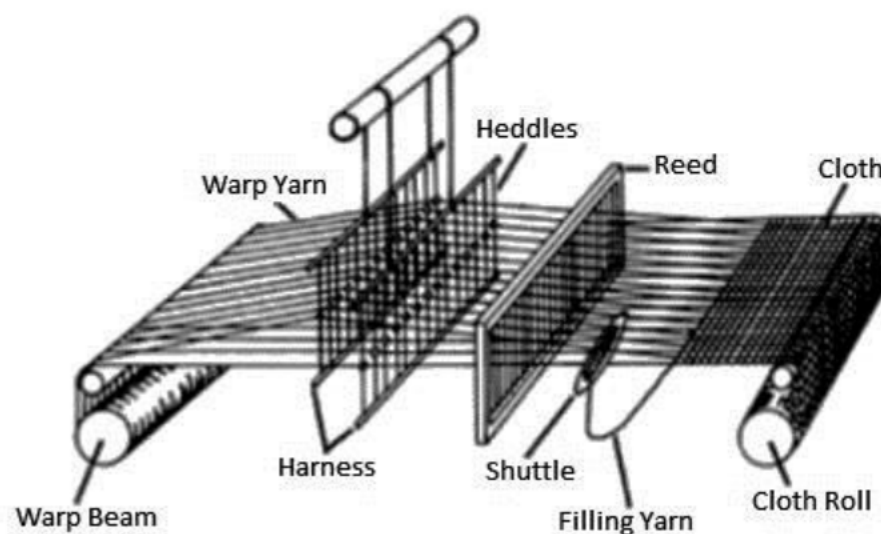


Figure 1-2: A simplified sketch of a two-harness loom [1, 2]

During the weaving process one harness is raised with the other left in the down position forming a V-like opening called shed. A filling yarn is then inserted in the shed and travels across the width of the loom, passing over some warp yarns and under other warp yarns. The reed, a comb-like device, then pushes the filling yarn into the body of the fabric. The woven fabric is rolled onto the cloth roll or fabric beam as it is produced. Those steps of raising one or more harnesses to separate the warp yarns and form a shed (shedding), inserting the filling

yarn through the shed (picking), pushing the filling yarn into place in the fabric with a reed (beating up) and winding the finished fabric onto the fabric beam (take-up) are repeated over and over again.

1.3 Importance of fabric defect detection and classification

Quality inspection of textile products is an important problem for fabric manufacturers. Good quality products increase profitability and customer satisfaction. Defects in textile fabrics affect the appearance and the integrity of the fabric products and that leads to loss of customer acceptance and confidence of the manufacturer. Therefore, good quality improves the manufacturer competitiveness in the global market. In addition, fabric defects lead to a direct loss in terms of money to the manufacturer. It has been estimated that the price of fabrics is reduced by 45%-65% due to the presence of defects [3]. It is also estimated that fabric defects are responsible for nearly 85% of the defects found by the garment industry [4].

Presently, most defects arising in the production process of a textile material are still detected by human inspection [5]. However, that way of visual inspection has several weaknesses including fatigue, boredom and inconsistent performance of human inspectors. The high cost of human labour is also to be mentioned. More importantly, human defect detection rate in fabrics is low. It has been reported that human inspectors do not detect more than 70% of fabric defects and cannot deal with fabrics wider than 2 m and moving faster than 30 m/min [6]. Yet, a typical web material is 1-3 m wide and is driven with speeds ranging from 20-200 m/min [6]. Therefore, automated detection of fabric defects, which would result in the production of high-quality products at a high production speed is definitely desirable.

The traditional procedure for fabric quality inspection is manual and is carried out offline by human inspectors. The fabric produced from the weaving machines is about 1.5-2 meters wide, and rolls out at the speed of 0.3-0.5 meters per minute. When a human inspector notices a defect on the moving fabric, he stops the motor that rotates the fabric roll, records the defect with its location and starts the motor again [7, 8].

1.4 The fabric defects and their classes

The fabric defect can be defined as an abnormality in or on the fabric structure. Fabric defects are caused by the yarn quality or/and malfunction during the weaving process. The yarn defects themselves may come from the fibre defects or a malfunction during the spinning process. Figure 1-3 shows a flowchart of woven fabric defects based on their source [9].

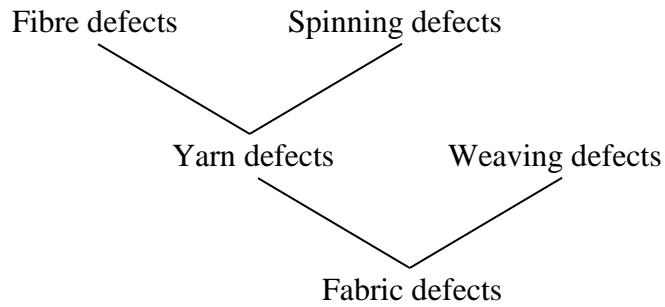


Figure 1-3: Flowchart of woven fabric defects based on their source [9]

Among the defects that may be due to the yarn quality one can name colour and width inconsistencies, hairiness, slubs, broken ends etc [10]. The weaving defects include broken end, broken pattern, double end, float, gout, hole or tear, missing end, oil or other stain etc [11]. According to an industry survey conducted by Lewis et al. [4], broken picks, harness drops and start marks top the list of the most frequently occurring defects, while broken ends, broken picks, waste and coarse picks are the most costly defects in terms of penalty points explained later in this section.

Table 1-1 lists the common defects in woven fabrics as well as their description.

The automatic classification of fabric defects into their categories is useful because the information about a defect category may reveal the origin of the defect. For example, a high occurrence rate of a certain type of defect may indicate malfunction of components of the weaving machine. This would allow maintenance technicians to locate the faulty component and repair it. Furthermore, statistics about each type of defects provide necessary information for the quality grading of the fabric [12].

Typically, the finished fabric is inspected for faults according to industry standards [13]. For example, in the standard four-point system of fabric inspection penalty points are given for detected defects. The amount of the penalty depends on the length of the defect as follows: 1 penalty point is given to defects of 3 inches or less, 2 penalty points are being given to defects of between 3 to 6 inches, 3 penalty points are given to defects of between 6 to 9 inches and 4 penalty points are given to defects of above 9 inches. Any defect of continuous nature is given 4 points for each yard in which it occurs. Severe defects such as hole are assigned the maximum 4 points for each yard in which they occur. The quality of the batch of cloth is described by the number of penalty points per 100 yards of inspected cloth. A penalty of up to 40 points is generally considered as an acceptable defect rate. Apart from the four-point system described above, other standards, such as the more complicated ten-point system or the Dallas System for knitted fabric, may be used to measure the quality of cloth.

Table 1-1: Common fabric defects [9]

Defect type	Definition	Reasons	Severity
Floats	A portion of a yarn in a fabric that extends or floats, unbound, over two or more adjacent ends or picks	It is caused by missing of interlacement of two series of threads.	Major fabric defect
Weft curling	A twisted weft thread appears on the surface of the fabric	It is caused by inserting a highly twisted weft thread or when the weft thread is running too freely.	Minor fabric defect
Slubs	A local uneven fabric thickness	It is caused by an extra piece of yarn that is woven into fabric. It can also be caused by thick places in the yarn or by fly waste being spun in yarn during the spinning process.	Minor/major fabric defect
Holes	A fabric area free of both of warp and weft threads	It is a mechanical fault caused by a broken machine part.	Major fabric defect
Oil stains	A fabric area contains oil spots	It is caused by too much oiling on loom parts or from other external sources.	Minor/major fabric defect
Stitching	A common fabric fault in which the ends and the picks are not interlaced according to the correct order of the pattern	As the main purpose of the loom is to interlace two sets of threads according to the correct order of the pattern, this defect is a result of any undesired motion of the main or auxiliary loom mechanisms such as: shedding, picking etc.	Major fabric defect
Rust stains/dirt	A dirty area or when fabric contains stains	Stains are caused by lubricants and rust. Most of the stains can be traced back to poor maintenance and material handling.	Minor/major fabric defect
Knots	A fabric place where two ends of yarn have been tied together and the tails of the knot are protruding from the surface	It is caused by tying spools of yarn ends together.	Minor fabric defect
Temple marks/pinholes	Marks or holes along fabric selvage	It is caused by the temples or pins which hold the fabric while it processes through tenter frame.	Minor fabric defect
Snag	A thread segment or group of fibres pulled from its normal pattern	It is created due to the friction between the fabric and sharp or rough objects.	Minor fabric defect
Tear	Damaged fabric portions differ from holes in that it has a random uneven shape	It is created due to the friction between the fabric and sharp or rough objects.	Major fabric defect

Table 1-1: Common fabric defects [9] (continued)

Defect type	Definition	Reasons	Severity
Gouts	A local uneven fabric thickness differs from slubs in that they are characterised by a lumpy appearance while slubs generally are symmetrical	It is caused by masses of accumulated short fibre (fly) being drawn undrafted into the filling yarn during the spinning process.	Major fabric defect
Weft snarls	A short length of three fold weft yarn of which two folds are inter-twisted	It is caused due to insufficient twist setting which increases the possibility of yarn severe rubbing between the shuttle and the box front plate	Minor fabric defect
Moiré	Presence of wavy areas in a periodical sequence, where crushed and uncrushed threads reflect light differently that affects the fabric appearance	It is caused due to a different compression of weft and/or warp threads.	Major fabric defect
Miss-end	A warp thread is absent in the fabric for a short or long distance	It is due to incorrect warping or by a broken warp thread that never replaced by another one.	Major fabric defect
Warp stripes	One or more faulty threads giving rise to zones of different aspect	It is caused by scraping or rubbing between warp threads and some parts of production machines or due to inaccurate reeding.	Major fabric defect
Tight/slack warp thread	A warp thread or pieces of warp thread which are tighter or slacker than the other pieces/threads	It is caused due to the incorrect tension applied on warp threads.	Major fabric defect
Double-ends	Two ends threaded in the same place as one	It is caused by incorrect warping or by a broken end wound on another and takes the behaviour of one thread.	Major fabric defect
Coarse-end	A warp thread or pieces of warp thread which are coarser than the other pieces/threads	It is caused due to the presence of a warp thread that has different count (coarser thread) than the other warp threads.	Major fabric defect
Smash	Many ends or warp threads are consequently broken	It is caused by a wrong timing of shedding, soft picking, insufficient checking of shuttle in the boxes, severe slough off, and damaged or broken picking accessories.	Major fabric defect
Open reed	It is conspicuous on fabrics that use different coloured threads on warp and weft where the warp threads is held apart, exposing the filling ones	It is caused due to the bent reed wires leaving a crack in the fabric.	Major fabric defect

Table 1-1: Common fabric defects [9] (continued)

Defect type	Definition	Reasons	Severity
Miss-pick	A weft thread is absent in the fabric for a short or long distance	It is caused by incorrect picking or if the weaver restarted the loom after any stoppage without adapting the position for the new insertion.	Minor/major fabric defect
Irregular pick density	A jammed or opened area formed in the fabric due to uneven pick density (number of picks per inch)	It is a mechanical fault caused by an irregular beating up force.	Major fabric defect
Double-picks	Two weft threads take the same place of one thread	It is caused by incorrect picking.	Major fabric defect
Coarse-pick	A weft thread or pieces of weft thread which are coarser than the other pieces/threads	It is caused due to the presence of a weft thread that has different count (coarser thread) than the other weft threads.	Major fabric defect
Starting mark (weft bars)	A visual light/dark effect in weft direction	It is caused by a higher or lower weft density caused by the weaving machine.	Major fabric defect
Tight/slack weft thread	A weft thread or pieces of weft thread which are tighter or slacker than the other pieces/threads	It is caused due to the incorrect tension applied on weft threads.	Major fabric defect
Skew/bias	When the weft threads are not square or perpendicular with warp threads	It is caused due to the variation of the beating up force value after the insertion of weft threads.	Minor fabric defect

1.5 Commercial automatic defect detection systems

There is little published information about existing commercial fabric defect detection and classification systems, probably because of trade secrecy. One such publication was provided by Alfred Dockery.

According to Dockery [14] fabric inspection has proven to be the most difficult of all textile processes to automate. However, some commercial automated optical fabric inspection systems do exist. Examples of such systems include Cyclops manufactured by BMS (Belgium Monitoring Systems, formerly Barco Vision), I-Text by Elbit Vision System and Fabriscan by Zellweger Uster.

Cyclops has a travelling scanning head and can be deployed on the weaving machine itself. It can prevent the production of off-quality fabric by stopping the weaving process if it detects a serious or running defect. After resolving the defect's cause the weaver makes a declaration on the loom terminal so that Cyclops releases the loom for further production. When

connected to the QualiMaster system (a system of fabric inspection and reporting also from BMS) all defect information, pick and time stamped, is sent to a fabric quality database. This allows producing defect maps and various types of quality reports [15].

The I-Text system is capable of inspection speeds up to 300 metres per minute and can handle fabric widths of up to 5 metres. The system's proprietary software learns the normal pattern of the fabric and detects changes. These changes in the pattern are then analysed by multiple detection algorithms to separate real defects from random but normal variations in the fabric. Once a defect is detected, the x-y location and the size of the defect are recorded in a defect map. In addition, a digital image of the defect is saved for later review by the system operator [14].

Fabriscan can inspect fabric at speeds of up to 120 meters per minute and can detect defects down to a resolution of 0.3 millimetres. It can handle fabric widths from 110 to 440 centimetres. Fabriscan classifies defects in a matrix called Uster Fabricclass. That matrix has two axes. The x-axis is for the length of the defect, while the y-axis is for the contrast of the defect. This allows the system to tell the difference between disturbing versus non-disturbing defects. Data on defects can also be stored in a relational database, allowing users to generate any type of report that they need [14].

1.6 Research in the field of fabric defect detection and classification

As shown on the block diagram of Figure 1-4, fabric inspection generally implies two stages; fabric defect detection and fabric defect classification. Fabric defect detection declares the presence or absence of a defect, while defect classification puts the detected defects into their corresponding predefined classes.

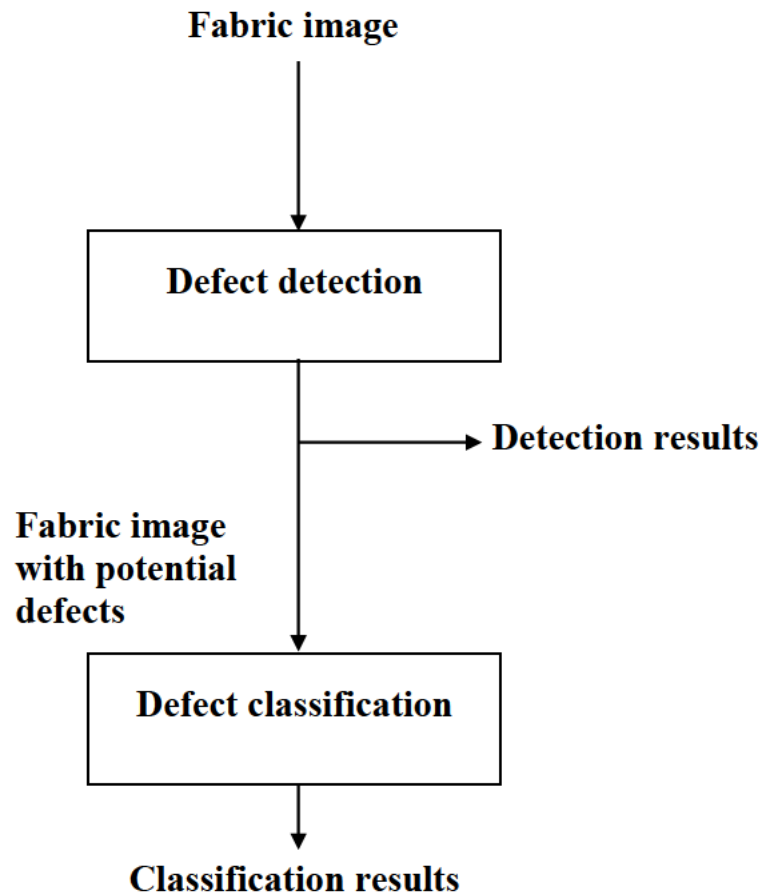


Figure 1-4: Block diagram of a fabric inspection system [16]

Figure 1-5 shows a more detailed block diagram. A fabric image is captured and then sent into a preprocessing stage. That stage is used to reduce noise from captured images and to eliminate the effect of uneven illumination. It can also be used to improve the contrast and the dynamic range of grey level intensity by using processing methods such as histogram equalisation.

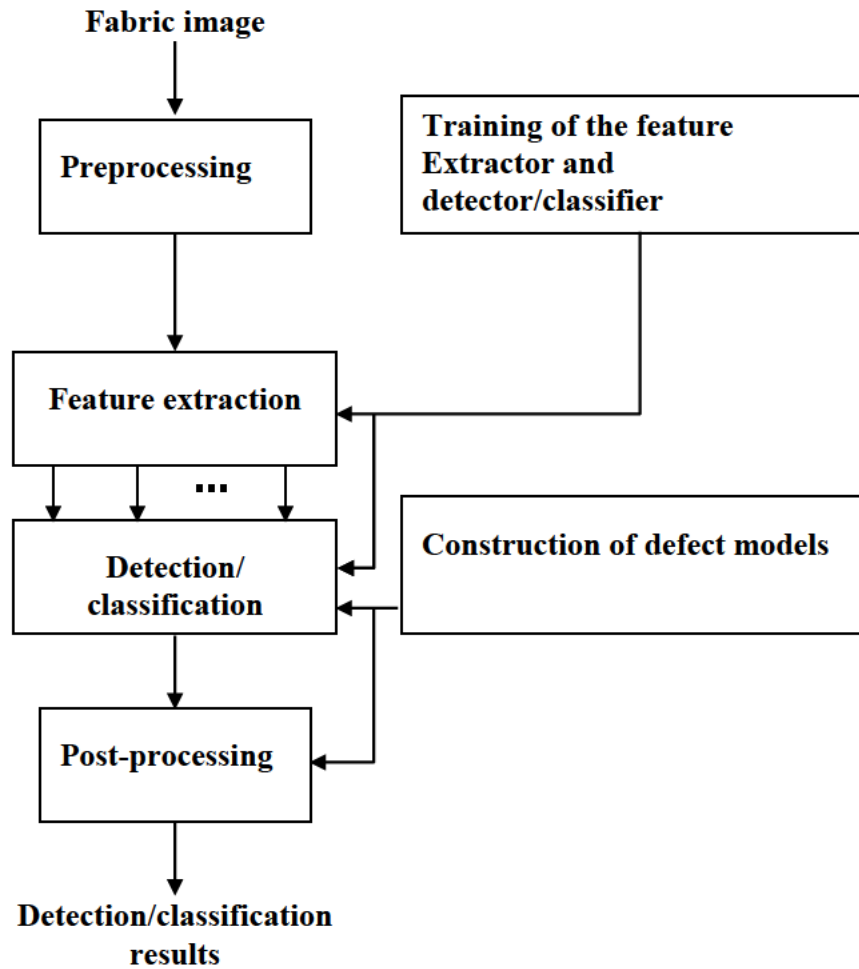


Figure 1-5: Detailed block diagram of a fabric inspection system [16]

The stage of feature extraction consists in calculating some texture descriptors of the image textile areas that are then fed into a classifier/detector. A defect detection or classification method is usually named after the feature extraction method used.

Various methods for automated detecting defects in textile fabrics have been proposed in literature. In his survey, Kumar classified the various methods into 3 main categories, namely statistical approaches, spectral approaches and model-based approaches [10].

Statistical texture analysis methods measure the spatial distribution of pixel values [17]. A large number of spatial texture features have been proposed, ranging from first order statistics to higher order statistics. Amongst others, histogram statistics, rank order approaches, co-occurrence matrices, autocorrelation and local binary patterns have been applied to fabric defect detection. Methods using low-order statistics (e.g. histogram-based analysis and rank order approaches) are usually simple and fast but are not as a rule sufficient guarantee of high

accuracy [18]. High-order statistics (e.g. co-occurrence matrices) provide much more accurate models, but are characterised by huge memory and computational requirements [19].

The usage of spectral features for the detection of defects is based on the high degree of periodicity of basic texture primitives, such as yarn in case of textile fabric. However, random textured images cannot be described in terms of primitives and displacement rules, as the distribution of grey levels in such images is rather stochastic. Therefore spectral approaches are not suitable for detection of defects in random textured materials. In spectral-domain approaches, the texture features are generally derived from the Fourier transform, Gabor transform and wavelet transform [20].

Through various publications spectral approaches, in particular Gabor filtering, have been more popularly applied in these areas. Spectral approaches are not suitable for detecting defects in random textured materials. Model-based approaches are more suitable for those randomly textured fabrics [17]. Markov random fields (MRF) have been popular for modelling images. Özdemir et al. [18] compared six texture features consisting of MRF, KL transform, 2D lattice filters, Law filters, co-occurrence matrices and a FFT based method. Texture modelling using a high (9th) order MRF model gave the best detection result.

Some effort is still needed in the field of fabric defect detection to increase the quality and performance of defect detection and classification systems. Authors such as Kumar suggest a combination of statistical, spectral and model-based approaches for further research to obtain better results than any one method individually [10].

There have been some attempts of combining wavelet transform with grey level co-occurrence matrix (GLCM) for feature extraction. This was generally done by extracting GLCM features from the wavelet transform sub-image for texture segmentation, which proved to be successful [21-23]. However, such methods did not compare the obtained detection performance with that obtained by using individual methods (GLCM or wavelet), nor did such methods compare the obtained performance with the detection performance that would be obtained by directly pooling together GLCM and wavelet features.

1.7 Outstanding problems (challenges) of fabric defect detection and classification

There has been a lot of research in the field of fabric defect detection and classification, but there are still obstacles. The major challenges in fabric defect detection and classification include [10, 16, 24, 25]:

1. There are numerous categories of fabrics [24].
2. The data throughput is enormous [16].

3. Great number of fabric defect classes; at present there are more than 70 classes of fabric defects [24] defined by the textile industry.
4. Defects within the same class may have different appearances in different fabric materials and from different factories [16].
5. The diversity within each class of defect and the similarity among different classes of defects [16].
6. Some classes of defects only have slight difference from non-defective fabric [16].
7. Unlike images of other materials such as steel and paper, textile images are rich in details [25].
8. The stochastic variations in scale stretch and skew of fabric texture/defects, predominantly due to the environment and the nature of weaving process [10].
9. Defects inspection is done on moving and vibrating fabric [16].

Even if many different fabric defect detection have been proposed in literature and different detection success rates have been reported, it is rather difficult to compare the detection success rates among them because the experiments have been generally performed on different datasets. It is desirable to have a common dataset that could be used by different researchers so that the methods that they propose can be easily compared to each other. To emphasise that, Mahajan et al. [17] stated

‘... due to lack of uniformity in the image database, performance evaluation and the nature of intended application, it is not prudent to explicitly declare the best available methods for fabric defect detection. Therefore, the effective performance evaluation requires careful selection of data sets along with its clear definition of scope’.

1.8 The objective of this thesis

The objective of this thesis is to devise new defect detection and classification methods by combining existing statistical, spectral and model-based methods for improved performance. It will not deal with details about image capturing, but will rather make use of an existing fabric image dataset and focus on texture feature extraction and classification.

More specifically, this thesis will look into the combination of co-occurrence matrix feature-based methods (from the category of statistical methods), wavelet-based methods (from the category of spectral methods) and Markov random field-based methods (from the category of model-based methods).

Pertaining to the classification stage, the thesis will compare the Euclidean distance, the K-nearest-neighbour and the feed-forward neural network classifiers.

1.9 The contribution of this thesis

Experiments involving various defect detection methods on a large, unique dataset allowing for the comparison of their performances.

Identification of the optimal number of GLCM features for fabric defect detection.

Three combinations schemes of fabric defect detection and classification methods involving statistical, spectral and model-based methods.

The use of complex wavelet transform to extract features that lead to improved performance in fabric defect detection and classification.

1.10 The outline of this thesis

Chapter 1 presents background information about textiles and fabric defect detection and classification. It also presents the objective, scope and contributions of this thesis.

Chapter 2 reviews the main published research in the area of fabric defect detection and classification. The methods proposed by various authors are grouped into three categories: statistical, spectral and model-based approaches.

Chapter 3 presents conceptual information about the methods that are used in the experimental part of the thesis. GLCM, wavelet transforms and Markov random fields are successively presented. Next, the classifiers that will be used are presented.

In chapter 4, we describe the experimental data. After the overall description of the types of fabrics as well as the types of defects that the Textile texture database developed by the Texture Analysis working group of the Deutsche Forschungsgemeinschaft in Germany (TILDA) dataset contains, we define our elementary experimental sample and then describe how the whole dataset was divided into elementary experimental samples. Next, we present our method of labelling each sample with its respective class defect. We finally describe how the samples were divided into two groups, namely the training and testing sets.

Chapter 5 experiments with the GLCM features. The objective is to find a small set of co-occurrence matrix features that allows one to best discriminate the fabric defects and find an algorithm that allows their fast extraction.

In chapter 6, we experiment with wavelet-based features. We are interested in the undecimated wavelet transform and the dual-tree complex wavelet transform due to their shift-invariance property. We compare their respective performance when applied to the

detection and classification of defects in our dataset. Then, their best features for discriminating the defects of fabric images in our dataset are identified.

Chapter 7 experiments with the Markov random field features. We deal with Gaussian Markov random field models. For each of the different types of fabrics in our dataset, we identify the model order that yields the best discriminating parameters for fabric defects.

In chapter 8, we propose combination schemes of features identified in chapters 5, 6 and 7. Those combination schemes constitute new defect detection and classification methods that are tested using our dataset.

The thesis is concluded in chapter 9. In that chapter we present the suggestions for future research work.

Chapter 2: Review of fabric defect detection literature

2.1 Introduction

This chapter aims at presenting the main approaches to fabric defect detection and classification that are found in literature, highlighting their advantages as well as their limitations.

The literature related to fabric defect detection and classification covers a large variety of methods that have been attempted. Kumar [10] and Ngan et al. [24] provided recent reviews of research in that field. Other reviews include [17, 18] and [26].

Most publications take the problem of fabric defect detection and classification as a texture analysis problem and assume that a fabric defective region has texture that is different from a sound fabric area. Therefore it becomes possible to detect fabric defects by identifying regions where texture is different from the dominant background texture. The categorisation of the various defect detection techniques is generally based on the texture feature extraction method used by those techniques. Kumar [10] classified the fabric defect detection techniques into three categories: statistical, spectral and model-based. Ngan et al. [24] extended the number of categories to seven, adding the learning, structural, hybrid and motif-based categories to the statistical, spectral and model-based approaches. We will adopt the classification by Kumar. Specifically, the object of this thesis is to propose defect detection and classification methods that combine methods from each of the three categories for improved detection or classification performance. Studies [10, 17] have suggested the combination of statistical, spectral and model-based methods for improved fabric defect detection and classification.

The remaining part of this chapter will be organised as follows. Sections 2.2 and 2.3 review the methods in the statistical category. Section 2.2 reviews methods that make use of first order statistics, while Section 2.3 deals with methods that utilise second order statistics of image pixels. Section 2.4 deals with spectral methods with special emphasis on wavelet-based approaches. Section 2.5 deals with model-based methods with a particular emphasis on the Markov random fields-based methods.

2.2 Methods based on first order statistics

Methods based on first order statistics are those that make use of statistics of individual image pixels such as mean, variance, skewness and kurtosis. Usually a fabric image is divided into blocks, some first order statistics are calculated from the image pixels of each block, and then

each block is classified as defective or defect-free, comparing the calculated statistics to the same statistics of defect-free blocks. The assumption made by such methods is that the first order statistics remain the same within the defect-free region, but change drastically from a defect-free to a defective region.

Elragal [27] combined first order statistical features and second order textural features for improved detection and classification rates. He used a fuzzy C-mean clustering and adaptive neural-fuzzy inference system (ANFIS) for classification. A similar approach was adopted by Balakrishnan et al. [28] to classify defects in denim fabrics into four classes. Using first order statistics features (mean, variance, skewness, kurtosis and entropy) they obtained a classification rate of 70%. To improve the classification rate they combined the first order classification features with second orders statistics features and got a classification rate of 93%.

In order to reduce the computational cost, Neubauer [29] used local histograms on a window grid to detect defects in textile fabrics. The histograms were calculated on 10x10 and 20x20 windows. The number of grey levels for histograms was limited to 8 to reduce the amount of data. Classification was done using a perceptron net trained by backpropagation.

Abouelela et al. [25] used local mean grey values and local variances to detect defects in textiles. After a preprocessing stage to eliminate inhomogeneity due to illumination and to filter out false defects, the fabric image was submitted to a smoothing operation and then local variances were calculated. Defects were detected as fabric areas with high local variances.

The advantage of methods based on first order statistics is their relative low computational complexity and thus the defect detection is relatively fast. However, the first order statistics and pixel-wise analysis are not able to efficiently define or model a texture [30]. Therefore, statistical texture analysis methods usually employ higher order statistics and neighbourhood properties of texture.

2.3 Methods based on second order statistics

Methods based on second order statistics deal with statistics of pairs of pixels related in some manner. The most commonly used second order statistics methods for texture analysis are grey level co-occurrence matrices (GLCM), autocorrelation function and grey level run length (GLRL) [30].

2.3.1 Grey level co-occurrence matrices

The grey level co-occurrence matrix of an image, originally described by Haralick et al. [31], is used to describe the statistics of neighbouring pairs of pixels separated by a given distance d in a given direction θ . Details of compilation of the co-occurrence matrix are given in Subsection 3.2.2. In general, four such matrices are used to describe different orientations in an image. More specifically, one co-occurrence matrix describes pixels that are adjacent to one another horizontally, P^0 . There are also co-occurrence matrices for the vertical direction and both diagonal directions called P^{90} , P^{45} and P^{135} respectively [32].

After the compilation of co-occurrence matrices, usually a set of textural features are derived from them and those features are used to obtain the texture description. The commonly used co-occurrence matrix features are energy (also called angular second moment), entropy, correlation, local homogeneity (also called inverse difference moment) and inertia (also called contrast). The definitions and details of calculation of those features and more other features are provided in Subsection 3.2.3.

Several researchers have used the co-occurrence matrix and its features to detect and classify defects in textile fabrics [33-38], to detect defects in texture from different sources [23-42] or to classify texture in general [43]. Clausi [44] made an extensive analysis of co-occurrence matrix features when applied to classifying images of Synthetic Aperture Radar (SAR) sea-ice images.

One of the common issues when using co-occurrence matrices and their features is the choice of the distance d and the direction θ necessary for their compilation. These choices are not always motivated, and the distance $d=1$ and four directions $\theta=0^\circ$, $\theta=45^\circ$, $\theta=90^\circ$ and $\theta=135^\circ$ are usually used [18, 21, 31, 37-38, 44]. However, some researchers found ways of improving the choice of the distance d or the directions θ .

Tsai et al. [33], for example, chose the distance d that maximised the angular second moment and minimised the contrast features in the weft and warp directions for co-occurrence matrices from images of defect-free fabrics. They assumed a periodic behaviour of grey level values of pixels in the weft and warp directions, and using the fact that periodicity in the direction θ leads to large values of diagonal elements of the co-occurrence matrix compiled for along that direction, they showed that the distance d that maximises the angular second moment and minimises the contrast is the period of the grey value level signal.

Bodnarova et al. [32, 34] proposed a method of improving the choice of both the distance d and the direction θ that makes use of a χ^2 significance test. A similar method was used by

Murino et al. [35]. The optimal choice of parameters using the method proposed by those authors depends on the specific feature to be extracted from the co-occurrence matrix. To derive the optimal parameters d and θ which would provide the maximum structural information from the feature of interest, they performed a χ^2 test on EFMs (elementary feature matrices) corresponding to the considered feature. For different values of displacement vector (d, θ) , a χ^2 score was computed on EFMs. The optimal displacement vector (d, θ) was chosen as the one that produced the maximum value of the χ^2 score for that particular feature.

Another issue with the co-occurrence matrix-based methods is the choice of specific features to be used in a particular defect detection or classification task. Haralick al. [31] proposed 14 features that can be extracted from each co-occurrence matrix and some more features have been proposed by others researchers [42, 45-47]. It has also been shown that many of those features are correlated among each other [48]. Therefore there should be a method of selecting a suitable subset of those features that need to be extracted. Furthermore, using a small number of effective features allows decreasing the computational burden to extract the required features.

As shown by numerous publications, the most frequently used features are contrast, energy, entropy, homogeneity and correlation, or a subset of them [33-47]. However, most of those publications do not justify the choice of the specific features that they use. Some authors performed feature selection procedures for their specific applications. For example, Gomez et al. [47] used the mutual information (MI) technique to rank and select the features extracted from breast ultrasound images using the minimum redundancy maximum relevance (mRMR) criterion. The selected features were then used to classify the breast lesions on the images.

The number of grey levels in the input image is also an important issue. The input image is generally coded using 8 bits per pixel, and therefore each pixel can have values from 0 to 255. However such a great number of grey levels increases the computation cost of the co-occurrence features. So, often the input image is quantised so that the co-occurrence matrices are compiled from an image that allows less grey levels. An additional benefit of using a reduced number of grey levels is reduction of noise [45-47], although that gain may not compensate the loss of information as a result of quantisation. Several authors [18, 23, 31-35, 42, 43, 49] selected 8, 16, 32 and 64 grey levels for their applications, but most of them did not justify their choice.

A few researchers performed studies of the effect of the number of grey levels on the effectiveness of the co-occurrence matrix features. For example, Clausi [44] analysed the

change of classification ability of various GLCM features as the number of grey levels increased. He found that most of the features had poorer classification power with a high number of grey levels while the opposite was expected. Using the dissimilarity and contrast features, coarse quantisation (less than 24 grey levels) produced low classification results, while quantisation with more than 24 grey levels provided quite consistent results with minimal variability. The entropy, uniformity and maximum probability features all had a strong decrease in classification accuracy with increasing number of grey levels. The inverse difference and the inverse difference moment features had also decreasing classification with an increasing number of grey levels.

Soh and Tsatsoulis [46] studied the effect of the number of grey levels on the effectiveness of the GLCM features from SAR sea ice images. In their study, they used 8, 16, 32, 64, 128 and 256 grey levels obtained using the uniform quantisation scheme. They found that 8 grey levels were inadequate to represent texture effectively. Furthermore, they showed that the dissimilarity between two samples of different textures does not change systematically with the number of grey levels. However, they found that for the same sample the GLCM results obtained using a pair of quantisation levels (i.e. number of grey levels) were more consistent as the number of quantisation levels increased. On his side Shohr [48] found that the uniformity and entropy GLCM features of sea ice radar images were sensitive to the number of grey levels and that the ice classes were more separable for 16 levels compared to 4 and 64 levels. Different findings were obtained by Gomez et al. [47], who found that the quantisation level did not impact the discrimination power of GLCM features extracted from breast ultrasound images

The main weaknesses of the co-occurrence-based approach to fabric defect detection are as follows:

- 1) It is computationally costly to determine the co-occurrence matrices and to compute features from them [50].
- 2) It has poor performance for textures constructed by large-sized primitives [24]. Therefore there is a requirement that the defective regions in a fabric be large in order to discriminate between defective texture properties and non-defective texture properties. Thus this method is not appropriate for small-sized defects.
- 3) The co-occurrence matrices are a highly redundant way of representing texture [32] and therefore means of reducing redundancy by using optimal parameters (d , θ) and by feature selection is highly desirable.

- 4) Haralick et al. [31] proposed fourteen features that can be extracted from each co-occurrence matrix, and some more features have been proposed by other researchers [42, 44-47]. Therefore there is a feature selection problem of choosing the most appropriate features from those that can be extracted.

2.3.2 Autocorrelation function

The autocorrelation function (ACF) of a digital image $f(x,y)$ is defined as [51]

$$C_{ff}(p, q) = \frac{MN}{(M-p)(N-q)} \frac{\sum_{i=1}^{M-p} \sum_{j=1}^{N-q} f(i, j) f(i+p, j+q)}{\sum_{i=1}^M \sum_{j=1}^N f^2(i, j)} \quad (2.1)$$

where M and N are the numbers of rows and columns in the image respectively. It measures the correlation between the image itself and the image translated by a displacement vector (p, q) . Textures with strong regularity will exhibit peaks and valleys in the autocorrelation function [52]. The autocorrelation function can also be used to assess the coarseness and fineness of the texture. The autocorrelation function of a coarse texture drops off slowly and vice versa [30].

A few authors attempted to use the autocorrelation function to detect defects in textile materials. For example, Chetverikov and Hanbury [53] defined texture defects in terms of regularity. They considered defects as regions of abruptly falling regularity. In their study, they quantified pattern regularity by evaluating the periodicity of the autocorrelation function and then analysed it to detect defects in texture images including fabric images from TILDA. Wood [54] used an autocorrelation function in two dimensions to describe the translational and rotational symmetry of an image of plain carpet.

The autocorrelation function is generally considered as unsuitable for random textures with irregularly arranged textural elements [52].

2.3.3 Grey level run length method

This method compiles the number of grey level runs of various lengths. A grey level run is defined as a set of linearly adjacent pixels in an image with the same grey level value [51]. The run length is the number of pixels within the run. A longer run length implies a coarser texture and vice versa; also, a more uniformly distributed run length implies a more random texture and vice versa [30]. The number of grey level run lengths is organised in grey level run length matrices $R(\theta)=[r(i,j|\theta)]$. The matrix element $r(i,j|\theta)$ specifies the number of times

that a run length i of grey level j appears in the image in the direction θ . Normally four grey level run length matrices are compiled from each image, for $\theta=0^\circ$, $\theta=45^\circ$, $\theta=90^\circ$ and $\theta=135^\circ$. Texture features are then computed from the grey level run length matrices (GRLM) and used for the purpose of classification.

Some research work made use of the grey level run length features for defect detection in textiles. Siew et al. [55] used features extracted using four second order statistical methods, including the grey level run length, to discriminate the degrees of wear in a wool carpet. Sardy et al. [56] used textural features from the neighbouring grey level dependence matrix (NGDM) and grey level run length matrix (GRLM) and a backpropagation-trained neural network to identify three fabric types and detect defects from them.

Similar to the other methods based on second order statistics, GRLMs represent texture better than methods based on first order statistics. However, computing GRLMs is computationally costly because run lengths should be computed for all grey levels and lengths, and for different directions. Moreover, the texture discrimination ability of GRLM features is low compared to the other well-known texture features. For example, Singh and Singh [57] compared eight texture feature extraction methods including GRLM, grey level co-occurrence matrix (GLCM) and autocorrelation function (ACF) on the same dataset and GRLM performance was the lowest with about 43% correct classification, much lower than ACF (76%) and GLCM (79%).

2.4 Spectral methods

Unlike the statistical methods that work directly on grey-scale values of the image, the spectral methods perform some filtering or transform operations on the raw fabric images and then extract features from the results of those operations for the detection or classification of defects. In this section we review the spectral methods based on the Fourier transform, then on the Gabor filters and finally on wavelet transforms. For the wavelet-transform based methods we separate the real wavelet transform and the complex wavelet transform.

2.4.1 Methods based on the Fourier transform

Faultless fabric is a repetitive and regular global texture and therefore Fourier transform can be applied to monitor the spectrum of the fabric and detect defects from it [58]. This method is based on the assumption that the defects will modify significantly the spectrum of the fabric image to enable sensing their presence. Among researchers who used this method, Castellini et al. [59] used the optical Fourier transform to monitor the fabric structure, while Wood et al. [54] used Fourier and associated transforms to characterise carpet patterns. Ravandi and

Toriumi [60] used Fourier transform analysis to measure fabric appearance. They also discussed the fabric surface characteristics of fill and warp yarns for plain weave cotton fabric. Tsai and Hsieh [61] used the Fourier transform and the Hough transform to suppress directional texture with periodic or almost periodic lines from images under inspection. They then detected the defects using double thresholding. More recently, Malek [9] used the discrete Fourier transform in a system to detect fabric defects online.

The main disadvantage of Fourier transform-based methods is the lack of spatial information in the Fourier transform that makes it impossible to spatially locate the detected defects.

2.4.2 Methods based on Gabor filters

One way to overcome the limitations of the Fourier transform based-methods is to use Gabor filters. In the spatial frequency domain, a Gabor filter can be interpreted as a windowed or short-time Fourier transform. Unlike the Fourier transform, which is a global frequency content analyser, a Gabor filter performs a local analysis and returns the frequency contents of the signal in the neighbourhood of a specific point. It can be tuned to a specific direction.

A 2D filter is characterised by its impulse response

$$G(x, y) = e^{-\pi \left[\left(\frac{x-x_0}{\sigma_x} \right)^2 + \left(\frac{y-y_0}{\sigma_y} \right)^2 \right]} e^{-2\pi j(x_0 u + y_0 v)} \quad (2.2)$$

or by its transfer function

$$G(u, v) = e^{-\pi \left[\frac{(u-u_0)^2}{\sigma_u^2} + \frac{(v-v_0)^2}{\sigma_v^2} \right]} e^{-2\pi j(x_0(u-u_0) + y_0(v-v_0))} \quad (2.3)$$

Where u_0 and v_0 are the filter central frequencies in x and y directions, σ_x and σ_y are the filter standard deviations in x and y directions, x_0 and y_0 are the horizontal and vertical displacements in the spatial domain.

Gabor filters are a traditional choice for obtaining localised frequency information. They offer the best simultaneous localisation of spatial and frequency information [62].

Gabor filters showed high performance as feature extractors for texture discrimination [63-65]. Levesque [66] used their ability to be tuned to a specific frequency band and orientation

of image features to successfully segment some real and artificial textures. The images with different textures were submitted to Gabor filtering and the magnitudes of the filter responses were used to locate the areas occupied by each texture. The magnitude of the channel output should be large when the texture exhibits the frequency and orientation characteristics to which the Gabor channel is tuned.

More specifically, Gabor filters have been successfully applied to fabric defect detection. For instance, Bodnarova et al. [67] used Gabor filters to detect defects in Jacquard fabrics. They optimised their detection scheme by designing Gabor filters that maximise the Fisher cost function of their responses when applied to non-defective texture. Escofet et al. [68] used a set of 4x4 Gabor filters for detection of local defects in textile materials with periodic regular texture. The filters corresponded to four frequency levels distributed in octaves and four orientations (horizontal, vertical and two diagonals). The magnitudes of filter responses were then used as inputs to the segmentation algorithms. A similar method was proposed by Kumar and Pang [69]. Their unsupervised method used a Gabor filter bank of eighteen asymmetric filters distributed at three scales and six orientations. They also proposed a supervised method that made use of a single Gabor filter to isolate a particular class of defects. The Gabor filter was selected automatically using a heuristic algorithm to optimally discriminate that particular class of defects. Finally they proposed a defect detection method that used only the imaginary Gabor function in a bid to reduce the computational cost. More recently Mak and Peng [70] proposed a method to solve the problem of Gabor filter parameter selection for the purpose of fabric defect detection. They proposed a system based on a Gabor wavelet network consisting of three real-valued filters: (i) two real-valued Gabor filters and (ii) one smoothing filter. Jing et al. [71] used features extracted using Gabor filters for defect detection and features obtained using the local binary patterns and Tamura methods for defect classification.

The main advantage of using Gabor filters lies in its multiscale and multidirectional ability, combined with the ability of capturing space-localised information about textures. That allows effective defect detection of textures. However, disadvantages include the fact that a typical Gabor filtering is either expensive to compute, is noninvertible or both [72]. That disadvantage can be overcome using the dual-tree complex wavelet transform as described later in Subsection 3.3.8. Another problem with methods based on Gabor filtering is that the outputs of filter banks are not mutually orthogonal and therefore may result in a significant correlation between texture features [22]. Gabor filters have two other main limitations. The maximum bandwidth of a Gabor filter is limited to approximately one octave, and Gabor filters are not optimal if one is seeking broad spectral information with maximal spatial

localisation [62]. To overcome those limitations log-Gabor filters have been proposed [73-76].

2.4.3 Methods based on wavelet transforms

Similar to Gabor filters, wavelet transforms provide a way to perform multiresolution analysis of signals and images, but at significantly lower computational cost [77]. Another advantage of wavelet transform over Gabor filtering is that the filters (low-pass and high-pass) used in the computation of the wavelet transform remain the same for the different scales, while the Gabor filtering requires filters of different parameters [22]. The wavelet transforms provide information about both the frequency and time (or spatial location) of a signal. The wavelet-based defect detection methods belong to the category of spectral methods and appear to be the most promising for fabric defect detection and classification [16].

The wavelet transform of a signal $f(t) \in L^2(\mathbb{R})$ is obtained by convolving the signal with a shifted and scaled version of the original mother wavelet as described by the equation (2.4) [78].

$$Wf(s, u) = \left\langle f, \psi_{s, u} \right\rangle = \frac{1}{\sqrt{s}} \int_{-\infty}^{+\infty} f(t) \psi^* \left(\frac{t-u}{s} \right) dt \quad (2.4)$$

Where $Wf(s, u)$ denotes the continuous wavelet transform of the signal $f(t)$ for the scale parameter s and time (position) parameter u , $*$ denotes the complex conjugate operator and $\langle \dots \rangle$ denotes inner product operation. The mother wavelet function $\psi(t) \in L^2(\mathbb{R})$ should satisfy the admissibility condition described by (2.5), where $\hat{\psi}(\omega)$ is the Fourier transform of $\psi(t)$.

$$C_\psi = \int_{-\infty}^{+\infty} \frac{|\hat{\psi}(\omega)|^2}{\omega} d\omega < \infty \quad (2.5)$$

The admissibility condition (2.5) imposes that the wavelet function must have a zero average as described by (2.6).

$$\int_{-\infty}^{+\infty} \psi(t) dt = 0 \quad (2.6)$$

The wavelet described by equation (2.4) with a continuous scale parameter s and continuous time parameter u is known as continuous wavelet transform (CWT) and it is highly redundant. The discrete wavelet transform (DWT) allows computing the transform with no redundancy by selecting just enough discrete scale parameter values and discrete time parameter values. Therefore the DWT corresponding to a CWT function $Wf(s, u)$ is obtained by sampling the

coordinates (s, u) on a grid. The process called dyadic sampling, in which the values of discrete scale as well as the corresponding sampling interval differ by a factor of 2, is common. This leads to a modified wavelet representation given by equation (2.7) where j and k are integers.

$$\psi_{j,k}(t) = \frac{1}{\sqrt{2^j}} \left(\frac{t - k \cdot 2^j}{2^j} \right) \quad (2.7)$$

Then the DWT of a function f(t) would be given by equation (2.8).

$$Wf(j, k) = \langle \psi_{j,k}, k \rangle = \int_{-\infty}^{+\infty} \psi_{j,k}(t) f(t) dt \quad (2.8)$$

More details on wavelet transforms are provided in Section 3.3.

Wavelet transforms have been used by numerous authors for the purpose of texture classification in general and fabric defect detection and classification in particular. Generally, the wavelet transform was used as either texture feature extractor or as texture suppressor. The use of wavelet transform as texture suppressor is based on the rationale that for a given texture, textural information is dominantly located in a limited band of frequencies. Removing that band of frequencies from the results of the wavelet transform and performing the inverse wavelet transform make it possible to effectively suppress that texture from the original image. When used as texture suppressor, non-texture based methods are used to complete the defect detection task.

For instance, Sari-Sarraf and Goddard [40] used the wavelet transform as a preprocessing tool to attenuate the background of fabric images and accentuate the defects. That was done by performing a M-level wavelet decomposition using the Daubechies' wavelet. The optimal number of decomposition level M was selected manually by observing the output for a handful of fabric images. After wavelet decomposition they submitted the 'details' (sub-images) to a fusion scheme. The fused output was then submitted to histogram equalisation and then measurements based on correlation dimension were extracted from the result of the histogram equalisation. The thresholding operation on the fused image based on the global homogeneity measure produced a binary image that was then submitted to a two-pass blob analysis. Once the blob analysis was completed, the segmented defects could be classified into meaningful classes based on a few extracted features such as size, orientation and intensity profile.

Tsai and Hsiao [79] also used the wavelet transform as texture suppressor and then applied binary thresholding to separate defects from the background. To suppress texture, the fabric image under test was decomposed at certain levels of wavelet transform and then the image was reconstructed from selected detail images and approximation image in order to eliminate all regular, repetitive textures in the reconstructed image. The number of decomposition levels and the decomposed sub-images used for image reconstruction were determined from a texture model. After experimenting with a few orthogonal wavelets, including Haar, Daubelets, Symmlets, and B-spline biorthogonal wavelets (BS2.2 and BS3.9), the authors concluded that orthogonal wavelets with compact support length such as D4 and S8 were preferable in order to avoid leaked subtle anomalies and have fast computation. They also found that the number of wavelet decomposition between three and four was generally sufficient to remove regular, repetitive textures and to enhance defects in the restored images.

Yang, Pang and Yung [80] adopted the undecimated wavelet decomposition and thresholding for defect detection. The coefficients of the wavelet filter were adjusted adaptively to the fabric texture during the training stage in order to maximise the ratio of wavelet transform energy between the defect area and the defect-free background at a certain scale. The separation of defects was done by thresholding on energy of wavelet coefficients and the threshold was selected as $\mu + 3.5\sigma$, where μ and σ are the mean and standard deviation of the energy of the same wavelet transform on defect-free area. The choice of the undecimated wavelet transform over the orthogonal wavelet transform was motivated by its translation invariance property and the more degrees of freedom it offers on the wavelet design with respect to the orthogonal wavelets. This method was further improved by designing a specific adaptive wavelet for each defect so that eight different adaptive wavelets were designed to classify defects into eight classes [81].

Other methods sought to optimise wavelet coefficients for defect detection by using a genetic algorithm [82, 83]. They used a genetic algorithm to find the wavelet filter coefficients that minimised entropy in the wavelet transform of images of woven fabrics. Minimising entropy in images tends to filter out fabric texture while highlighting fabric defects. After obtaining the optimised wavelet filter coefficients, they used them to perform a wavelet transform on fabric images. They then isolated fabric defects by submitting the wavelet sub-images to double thresholding.

Ralló et al. [84] used Gabor wavelets and binary thresholding for detecting defects in textile fabrics. The image was submitted to a four-level wavelet decomposition using the complex Gabor wavelet. Ngan et al. [85] combined the wavelet transform method and a method they called golden image subtraction (GIS) for defect detection on pattern textiles.

Han and Shi [86] combined the wavelet transform and grey level co-occurrence matrix (GLCM) methods. The input fabric image was decomposed at various levels by wavelet transform. After each level of decomposition, the GLCM of the approximation sub-image was compiled. Then the homogeneity feature was calculated from the GLCM. By analysing the homogeneity features at different levels of wavelet decomposition, the appropriate level at which the approximation sub-image can be reconstructed into a non-texture image was determined. Once the non-texture image was reconstructed, they used the Otsu thresholding as detection technique, but also used the mean shift segmentation for complicated images where the thresholding techniques were invalid.

When used for feature extraction normally the texture image is submitted to wavelet transform operation, and then the features are calculated from the ‘detail’ and/or ‘approximation’ wavelet coefficients. The calculated features are then fed into a classifier for decision making.

One of the main issues concerning that use of the wavelet transform for texture characterisation is the choice of the wavelet to be used. Generally wavelets are characterised by properties such as orthogonality, size of support, regularity, symmetry, degree of shift variance, directionality in two and higher dimensions, and the number of vanishing moments, which could possibly influence their texture discrimination ability and therefore their performance in detecting defects in textile fabrics. However, few authors motivated the choice of the wavelet they used in their publications on defect detection or classification. Yang, Pang and Yung [12, 87-88] used the undecimated discrete wavelet transform because it is shift-invariant and allows more flexibility in design than the orthogonal wavelet. Karras et al. [89] extracted features from detail wavelet coefficients for the purpose of defect detection in textile fabrics. They tried several wavelet bases including Haar, Daubechies, Coiflet, Symmlet etc., as well as with Meyer's and Kolaczyk's wavelet transforms. However, only the Haar wavelet transform exhibited the expected and desired properties. All the other orthonormal, continuous and compactly supported wavelet bases smoothed the images so much that the defective areas did not appear in the sub bands.

For texture classification in general Mojsilovic et al. [90] suggested that the degree of shift variance is more important than regularity, with wavelets with low degree shift-variance performing better. They also suggested that regularity, number of vanishing moments and overall quality of high-pass filters do not have much influence on texture discrimination. Ahuja et al. [91] studied the properties determining the choice of mother wavelet in different applications and proposed the B-spline wavelet family for wavelet-image sequence super resolution.

The other issue relating to the use of wavelet transform for texture feature extraction is the choice of the adequate number of levels of wavelet decomposition. Also, the fact that each level of wavelet decomposition produces four sub bands leads to the problem of choice among the sub bands. Generally, the more is the number of wavelet decomposition levels, the more is the information in the extracted features but the more is the computational cost. The number of decomposition levels used by numerous authors varies from one to five [92-97].

Important in fabric defect detection and other pattern recognition applications of wavelet transform is the issue of shift-variance. That refers to the fact that the same texture pattern in the original image may lead to different wavelet coefficients depending on the pattern location. Wavelet bases as implemented by the DWT algorithm [78] are shift variant and therefore not suitable for pattern recognition applications such as fabric defect detection. Shift-variance is introduced by the down sampling process that is part of the DWT algorithm. Various authors have dealt with that problem in different ways. For instance, Sari-Sarraf and Goddard [40] used a variant of wavelet transform called multiscale wavelet representation (MSWAR) which is shift-invariant. Yang, Pang and Yung [12, 87-88] used the undecimated discrete wavelet transform which is also shift-invariant. More recently, Wang et al. [98] used the 2D dual-tree complex wavelet transform which is almost shift invariant and has other interesting properties as described in Subsection 3.3.8.

2.4.4 Methods based on the dual-tree complex wavelet transform

As mentioned previously, the discrete wavelet transform (DWT) in its critically-sampled form suffers from the problem of shift variance that makes it unsuitable for pattern recognition applications such as fabric defect detection [78]. The solution that has been usually used is the undecimated discrete wavelet transform (UDWT) [12, 87-88].

Even if the UDWT solves the problem of shift variance, it has a high redundancy rate of $3L+1$ for image representation, where L is the number of wavelet decomposition levels. That leads to increased computational requirements [99]. In addition, it does not solve the shortcoming of poor directional selectivity for diagonal features in 2D. This weakness of UDWT lowers the discrimination power of its texture features.

The dual-tree complex wavelet transform (DTCWT), first introduced by Kingsbury in 1998 [99-100], is approximately shift invariant and makes it possible to get directional wavelets in two and higher dimensions with only $2x$ redundancy in 1-D (2^d for d -dimensional signals, in general) [72]. It has been used in several research papers, especially for texture characterisation. For example, Costin and Ignat [101] discussed the effectiveness of the cosine similarity measure, the Pearson coefficient as well as the Frobenius norm applied to

the magnitudes of DTCWT coefficients of texture images in order to decide on their similarities. Other applications include content-based image retrieval [102-103], image segmentation [104] and texture classification [105-106].

Specific to textiles, Wang et al. [98] used the 2D dual-tree complex wavelet transform to separate the fabric texture and the pilling information from the image of the pilled nonwoven fabric. They then used that pilling information and a supervised neural network classifier for objective pilling evaluation.

2.5 Model-based methods

Model-based approaches seek to represent texture by a stochastic model. The model should be able to capture enough texture characteristics so that discriminating different textures based on the model becomes possible. The broadly used model for textured images representation is the Markov random field (MRF), introduced to the image processing community by Geman and Geman [107] in 1984.

Markov random fields modelling assumes that the intensity at each pixel depends only on the intensity of neighbouring pixels. That is expressed by a conditional probability density described by (2.9).

$$p(X(i)|Neighbours\ of\ i) \quad (2.9)$$

where $X(i)$ is the pixel intensity at location i . According to the Hammersley-Clifford theorem [108], a MRF, expressed by its local conditional probability, is equivalent to a Gibbs random field, expressed by the joint probability described by (2.10)

$$p(X) = \frac{1}{Z} e^{-\frac{1}{T} U(X)} \quad (2.10)$$

where X represents the joint event of all the pixels of the image, $U(X)$ represents the energy function, T is a constant called temperature and Z is a normalising constant called partition function [109]. Examples of types of Markov random field models are the auto-logistic, auto-binomial, multilevel logistic and auto-normal Markov random field models [109]. The auto-normal MRF models are also called Gaussian Markov random fields (GMRF).

In MRF modelling a texture is assumed to be a realisation of an MRF, and modelling a texture is equivalent to specifying the corresponding conditional probabilities or Gibbs model. Texture features correspond to the MRF model parameters and feature extraction is equivalent to parameter estimation [110].

The Markov random field models have been used for texture analysis and fabric defect detection. For example, Chellappa and Chatterjee [111] used GMRF models to classify textures from the Brodatz album, while Cross and Jain [112] used the auto-binomial models to analyse and synthesise textures. Wang and Liu [110] proposed a multi resolution MRF models for texture classification.

Cohen et al. [113] used GMRF to detect and locate the various kinds of defects that might be present in a given fabric sample based on its image. Because of the high computational cost of estimating the model parameters, they instead used the sufficient statistics associated with the models parameters which are easily computable. On their side Özdemir and Erçil [8] studied a model-based approach with MRF as texture model for the defect inspection of textile fabrics. They illustrated the results on real fabric images and implemented their method on a two TMS320C40-based parallel processing system.

Al-Kadi [114] combined statistical and model based features to classify eight different texture images. The statistical methods were GLCM, GLRL and ACF (autocorrelation function) while the model-based methods were GMRF and fractional Brownian motion. They found that when each method was used individually GLCM performed the best. However, when the different methods were combined in pairs, the model-based approaches used together achieved the highest classification rate compared to each of the five methods individually and to each other combination.

Compared to co-occurrence matrix-based approach, the model-based one is computationally more efficient. Similar to the co-occurrence matrix-based method, the texture model-based approaches are poor in discriminating small local defects, since a sufficiently large region is needed for a good estimation of the model parameters [16].

It is expected that MRF-based approaches perform better in fabric defect detection, since MRF models carry more local texture information than co-occurrence matrices [16]. In fact, Özdemir et al. [18] compared six texture features consisting of MRF, KL transform, 2D lattice filters, Law filters, co-occurrence matrices and a FFT-based method and texture modelling using a high order (9th) MRF model gave the best detection result. However, these results seem to contradict the findings by Clausi and Yue [115] when applied to remote sensed ice images. They compared co-occurrence matrix-based features and MRF features for discriminating textures in remote sensed ice images and found that co-occurrence matrix-based features had improved discriminating ability relative to MRF features with decreasing window size. Those findings are consistent with our own results as shown in chapter 8.

2.6 Summary

In this chapter, we reviewed various defect detection methods found in literature. Those methods are generally categorised on the basis of the texture feature extraction techniques that they use. They have been classified into three main categories: (i) statistical, (ii) spectral and (iii) model-based methods.

Statistical methods deal directly with statistics of grey level values of fabric image pixels. Among the statistical methods, one can distinguish those based on first order statistics from those based on second or higher order statistics. First order statistics methods deal with statistics of individual pixels. They are computationally simple, but they can detect only defects with a high contrast with respect to the defectless fabric background.

Methods based on second order statistics deal with joint statistics of pairs of pixels. Among the most commonly used one can mention (i) the grey level co-occurrence matrix, (ii) the autocorrelation function and (iii) the grey level run length method. Second order statistics methods model better texture than first order statistics methods and they provide better defect detection performance. However, they suffer from some weaknesses that include the high computational cost incurred for their use.

Spectral methods perform some filtering or transform operations on the raw fabric images before extraction features from the results of those operations. Several spectral methods exist. Among them we reviewed (i) those based on the Fourier transform, (ii) those based on Gabor filters and (iii) those based on the wavelet transform.

The Fourier transform allows characterising the fabric texture in frequency domain and detecting fabric defects by monitoring its spectrum. However, the Fourier transform lacks spatial information and therefore makes the methods based on it unable to locate spatially the detected defects.

Unlike Fourier transforms, Gabor filters offer simultaneous localisation of spatial and frequency information. They have been applied successfully to fabric defect detection. They show high performance as feature extractors for texture discrimination due to their multiscale and multidirectional ability, combined with their capability of capturing space-localised information about texture. However, typical Gabor filtering is either expensive to compute, is noninvertible or both.

Similar to Gabor filters, wavelet transforms provide a way to perform multi resolution analysis of signals and images but at significantly lower computational cost. Wavelet transforms also provide information about both time (or spatial location) and frequency of a

signal. They have been used for fabric defect detection as either texture suppressors or as texture feature extractors.

Numerous issues about wavelets need attention before their application for texture feature extraction. Among them is the choice of wavelet to use, the choice of the adequate number of decomposition levels and dealing with the problem of shift-variance of the wavelet transform.

The problem of shift-variance should be solved for pattern recognition applications of wavelet transforms such as fabric defect detection. The most commonly used wavelet transform, called the discrete wavelet transform (DWT), is not shift-invariant. The solution that is traditionally used is the undecimated discrete wavelet transform (UDWT). However, UDWT is highly redundant and computationally expensive. The better solution is the dual-tree complex wavelet transform (DTCWT), which, in addition to being almost shift-invariant with a low rate of redundancy, offers high directional selectivity in two and higher dimensions.

Model-based methods seek to represent texture by a stochastic model. One of the most used models for texture is the Markov random field (MRF) model. Texture feature extraction using a MRF model consists in choosing an appropriate model for the texture and then estimating the model parameters for that particular texture.

Chapter 3: Review of the methods employed in this thesis

3.1 Introduction

This chapter will present in detail conceptual information of the methods that will be employed in the experimental part of this thesis. The objective of the thesis is to combine grey level co-occurrence matrix, wavelet-based and Markov random field-based methods for improved fabric defect detection and classification. Therefore each of the three methods will be successively presented. These three tools are used for texture feature extraction. Once the features are extracted, they are fed into classifiers for decision about the presence of a defect or for categorising defects into predefined classes. Classifiers that will be used in this thesis will then be presented.

3.2 Grey level co-occurrence matrix features

3.2.1 Definition of grey level co-occurrence matrix

The grey level co-occurrence matrix (GLCM) analysis was introduced by Haralick et al. [31] in 1973. It is a second order statistics method of texture analysis in the sense that it is based on the computation of statistics of pairs of neighbouring pixels in an image, separated by a given distance d in a given direction θ .

A co-occurrence matrix is a square matrix whose elements are the number of occurrence of pairs of grey levels separated by the distance d in the direction θ . For an image with G grey levels, the grey level co-occurrence matrix P is a $G \times G$ matrix and its element $P_{d,\theta}(p, q)$ can be expressed by (3.11).

$$P_{d,\theta}(p, q) = \#\{(j, k), (m, n) : f(j, k) = p, f(m, n) = q\} \quad (3.11)$$

where $0 \leq p, q \leq (G - 1)$ are the grey level values of the pixels, the symbol $\#\{\}$ denotes cardinality of a set. After computing all elements of the matrix, each of them is divided by their total sum to normalise the co-occurrence matrix. The obtained normalised co-occurrence matrix therefore represents the joint probability of pairs of pixels having certain values.

Combinations of parameters d and θ allow to obtain different GLCM matrices from a single image. Generally, four directions, $\theta=0^\circ$, $\theta=45^\circ$, $\theta=90^\circ$ and $\theta=135^\circ$, and different distances (d) have been used in literature [18, 21, 31, 37-38, 45].

3.2.2 Compilation of a co-occurrence feature of a grey level image

Figure 3-1 shows the compilation of the grey level co-occurrence matrix of a simple 4x4 image with four grey levels for $d=1$ and $\theta=0^\circ$. A vector of length d and orientation angle θ scans the image from the top-left to the bottom-right of the image. The vector is positioned so that both its origin and end remain within the image. When the vector is positioned so that its origin is on a pixel with grey level value p and its end is on a pixel with a grey level value q , the co-occurrence matrix element (p, q) is incremented by 1. If the desired co-occurrence matrix is symmetric, the co-occurrence matrix element (q, p) is also incremented.

This is illustrated in Figure 3-1 by (a) and (b). In this case the vector is horizontal ($\theta=0^\circ$) and its length is 1. It is positioned on pixels with grey levels 0 and 3, so the matrix elements $(0, 3)$ and $(3, 0)$ are incremented. In this case, the GLCM is symmetric.

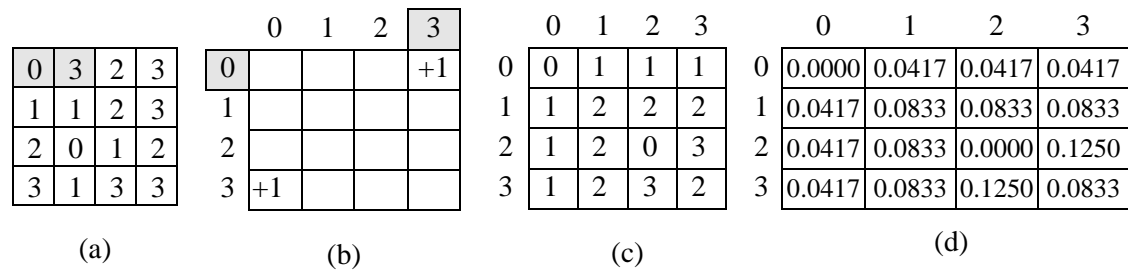


Figure 3-1: Construction of the co-occurrence matrix P^0 for $d=1$ [116]. The original image (a) begins by having each of its neighbouring pairs examined. Image (b) shows the incremental stage that occurs when the outlined neighbouring pixels in (a) are examined. Image (c) shows the result of the horizontal co-occurrence matrix for $d=1$, and (d) shows the final normalised co-occurrence matrix after dividing each element in (c) by 24, the sum of elements in (c).

When the scanning vector has occupied all the possible positions, the compilation of the co-occurrence matrix is completed. That corresponds to (c) in our illustrative example of Figure 3-1.

Finally, the raw co-occurrence matrix is normalised by dividing each of its elements by their total. The final normalised co-occurrence matrix contains the co-occurrence probabilities which are joint probabilities of pairs of pixels having certain values. That is illustrated by (d) in our example.

3.2.3 Co-occurrence matrix features

To characterise texture, usually some statistics, which are called co-occurrence matrix features, are computed from the normalised GLCM. Haralick et al. [31] proposed fourteen such features, while some more have been proposed by other researchers [42, 44-47].

In equations (3.12) through (3.42), $p(i, j)$ represents the $(i, j)^{\text{th}}$ entry of the GLCM, and G represents the number of grey levels in the image. In the equations defining the features we will make use of the means (μ_x and μ_y) and standard deviations (σ_x and σ_y) for the rows and columns of the matrix that are given by (3.12) through (3.15).

$$\mu_x = \sum_{i=0}^{G-1} \sum_{j=0}^{G-1} i p(i, j) \quad (3.12)$$

$$\mu_y = \sum_{i=0}^{G-1} \sum_{j=0}^{G-1} j p(i, j) \quad (3.13)$$

$$\sigma_x = \sum_{i=0}^{G-1} \sum_{j=0}^{G-1} (i - \mu_x)^2 p(i, j) \quad (3.14)$$

$$\sigma_y = \sum_{i=0}^{G-1} \sum_{j=0}^{G-1} (j - \mu_y)^2 p(i, j) \quad (3.15)$$

We will also make use of the quantities p_x , p_y , p_{x+y} and p_{x-y} defined by (3.16) through (3.19).

$$p_x(i) = \sum_{j=0}^{G-1} p(i, j) \quad (3.16)$$

$$p_y(j) = \sum_{i=0}^{G-1} p(i, j) \quad (3.17)$$

$$p_{x+y}(k) = \sum_{i=0}^{G-1} \sum_{j=0}^{G-1} p(i, j), i+j=k, k=0, 1, \dots, 2G-2 \quad (3.18)$$

$$p_{x-y}(k) = \sum_{i=0}^{G-1} \sum_{j=0}^{G-1} p(i, j), |i-j|=k, k=0, 1, \dots, G-1 \quad (3.19)$$

The fourteen GLCM features proposed by Haralick et al. [31] are as follows.

1. **Angular second moment**, also called **energy**. It is defined by (3.20). That feature is a measure of the homogeneity of the image. In a homogeneous image there are very few grey tone transitions. Therefore the matrix P will have few entries of large magnitudes. The ASM

(f_1) will be relatively high. On the other hand, a non-homogeneous image will have more grey tone transitions. The matrix P will have a larger number of small entries, and hence the ASM (f_1) will be smaller.

$$f_1 = \sum_{i=0}^{G-1} \sum_{j=0}^{G-1} (p(i, j))^2 \quad (3.20)$$

2. **Contrast**, also called **inertia**. It defined by (3.21). It is a measure of the contrast or the amount of local variations present in the image.

$$f_2 = \sum_{i=0}^{G-1} \sum_{j=0}^{G-1} (i - j)^2 p(i, j) \quad (3.21)$$

3. **Correlation**. It is defined by (3.22). It is a measure of grey tone dependencies in the image.

$$f_3 = \frac{\sum_{i=0}^{G-1} \sum_{j=0}^{G-1} (ij) p(i, j) - \mu_x \mu_y}{\sigma_x \sigma_y} \quad (3.22)$$

where μ_x , μ_y , σ_x and σ_y are the means and standard deviations of p_x and p_y .

4. **Sum of squares: variance**. It is defined by (3.23).

$$f_4 = \sum_{i=0}^{G-1} \sum_{j=0}^{G-1} (i - \mu)^2 p(i, j) \quad (3.23)$$

5. **Inverse difference moment**, also called **homogeneity**. It is defined by (3.24).

$$f_5 = \sum_{i=0}^{G-1} \sum_{j=0}^{G-1} \frac{1}{1 + (i - j)^2} p(i, j) \quad (3.24)$$

6. **Sum average**. It is defined by (3.25).

$$f_6 = \sum_{i=2}^{2G} i p_{x+y}(i) \quad (3.25)$$

7. **Sum variance**. It is defined by (3.26).

$$f_7 = \sum_{i=2}^{2G} (i - f_6)^2 p_{x+y}(i) \quad (3.26)$$

8. **Sum entropy**. It is defined by (3.27).

$$f_8 = -\sum_{i=2}^{2G} p_{x+y}(i) \log(p_{x+y}(i)) \quad (3.27)$$

9. **Entropy.** It is defined by (3.28).

$$f_9 = -\sum_{i=0}^{G-1} \sum_{j=0}^{G-1} p(i, j) \log(p(i, j)) \quad (3.28)$$

10. **Difference variance.** It is defined by (3.29).

$$f_{10} = \text{Variance of } p_{x-y} \quad (3.29)$$

11. **Difference entropy.** It is defined by (3.30).

$$f_{11} = -\sum_{i=0}^{G-1} p_{x-y}(i) \log(p_{x-y}(i)) \quad (3.30)$$

12., 13. **Information measures of correlation.** They are defined by (3.31) and (3.32).

$$f_{12} = \frac{HXY - HXY1}{\max(HX, HY)} \quad (3.31)$$

$$f_{13} = (1 - \exp(-2.0(HXY2 - HXY)))^{1/2} \quad (3.32)$$

Where HX and HY are entropies of p_x and p_y , and HXY, HXY1 and HXY2 are given by (3.33), (3.34) and (3.35) respectively.

$$HXY = -\sum_{i=0}^{G-1} \sum_{j=0}^{G-1} p(i, j) \log(p(i, j)) \quad (3.33)$$

$$HXY1 = -\sum_{i=0}^{G-1} \sum_{j=0}^{G-1} p(i, j) \log(p_x(i) p_y(j)) \quad (3.34)$$

$$HXY1 = -\sum_{i=0}^{G-1} \sum_{j=0}^{G-1} p_x(i) p_y(j) \log(p_x(i) p_y(j)) \quad (3.35)$$

14. **Maximum correlation coefficient.** It is defined by (3.36).

$$f_{14} = (\text{Second largest eigenvalue of } Q)^{1/2} \quad (3.36)$$

Where Q is a matrix given by (3.37).

$$Q(i, j) = \sum_{k=0}^{G-1} \frac{p(i, k)p(j, k)}{p_x(i)p_y(k)} \quad (3.37)$$

Some of the other GLCM features found in literature are the **dissimilarity** [45-47] defined by (3.38), the **cluster shade** and **cluster prominence** features [42, 46, 47] defined by (3.39) and (3.40), and the **maximum probability** and the **inverse difference** [45-47] defined by (3.41) and (3.42).

$$Dissimilarity = \sum_{i=0}^{G-1} \sum_{j=0}^{G-1} |i - j| p(i, j) \quad (3.38)$$

$$Cluster\ shade = \sum_{i=0}^{G-1} \sum_{j=0}^{G-1} (i + j - \mu_x - \mu_y)^3 p(i, j) \quad (3.39)$$

$$Cluster\ prominence = \sum_{i=0}^{G-1} \sum_{j=0}^{G-1} (i + j - \mu_x - \mu_y)^4 p(i, j) \quad (3.40)$$

$$Maximum\ probability = \max(p(i, j)) \text{ for all } (i, j) \quad (3.41)$$

$$Inverse\ difference = \sum_{i=0}^{G-1} \sum_{j=0}^{G-1} \frac{1}{1 + |i - j|} p(i, j) \quad (3.42)$$

3.2.4 Methods to improve the GLCM-related computational cost

One of the main weaknesses of GLCM features is the high computational cost incurred in compiling the co-occurrence matrix, as well as extracting features from it [50, 117, 118]. Some researchers have proposed ways to alleviate that problem.

The first solution taken by most researchers is the reduction of the number of grey levels in the image by quantising the original image. Several authors [18, 23, 31-35, 42, 43, 49] reduced the number of grey levels to 8, 16, 32 and 64 for their applications from 256 of their original images. That allows decreasing the computational cost as the size of the GLCM and therefore the required computational time increases with the square of the number of grey levels in the image.

Miyamoto and Merryman [116] proposed a solution that optimises the code. This was done in two steps: (i) optimising the code for the construction of the co-occurrence matrix using techniques of recursive blocking algorithm, scalar replacement and unrolling, and (ii) optimising the feature calculation by combining loops for different features into one loop, by scalar replacement and by using log tables for probability values (0 to 1) instead of computing them. Their solution allowed increasing the system performance by a factor of approximately two.

Tahir et al. [117] proposed a hardware solution. They used reconfigurable hardware – a field programmable gate array (FPGA) co-processor – to accelerate the calculation of the GLCMs as well as the extraction of features from them. The proposed implementation took advantage of the inherent parallelism in the calculation of GLCMs, as well as their features. The performance of the FPGA implementation was five times faster than that of a Pentium 4 PC.

Other researchers proposed solutions that made use of different representations of information in GLCM to speed up the calculation of features. Their rationale is that usually the GLCM is a highly sparse matrix, and therefore directly working on the matrix representation to extract features wastes a lot of time by adding and/or multiplying by zero values.

In that regard, Clausi and Jernigan [118] proposed a method they called grey level co-occurrence linked list (GLCLL) where only nonzero grey level probabilities were stored in a sorted linked list. Their method was improved by Svolos and Todd-Pokropek [119], who represented the same information in a tree data structure. Since GLCLL requires maintaining a sorted list, Clausi and Zhao [120] dropped the use of the sorted linked list and used instead the combination of a hash table and a linked data structure. They called the new improved method grey level co-occurrence hybrid structure (GLCHS). They then combined GLCHS and the grey level co-occurrence hybrid histogram (GLCHH) to get an improved method that they called grey level co-occurrence integrated algorithm (GLCIA) [121]. GLCHH is an implementation of the sum and difference histograms method proposed by Unser [122] in the GLCHS framework [121].

The computational improvement obtained from adopting those methods can be read from [121]. Given a variety of window sizes and quantisation levels, the GLCLL method required 0.20% to 18% of the computing time relative to the GLCM. The GLCHS is an improvement of the GLCLL, requiring 28% to 38% of the GLCLL computation time and 0.10% to 32% of the GLCM computation time depending on the quantisation and window size. Finally, experiments indicated that GLCIA requires 27% to 54% of the computation time compared to using GLCHS alone. The GLCIA computational time relative to that of the standard GLCM method ranges from 0.04% to 16% depending on the window size, quantisation and the features selected.

In Section 5.8 of this thesis we will propose a modified GLCHS method for implementation with MATLAB.

3.3 Wavelet transform and wavelet transform features

3.3.1 Introduction to wavelet transform

A wavelet is a ‘small wave’ which has its energy concentrated in time. It gives a tool for analysis of non-stationary signals such as short-time phenomena (wavelets are able to determine if a quick transient exists in a signal and if so localise it) [123]. The traditional signal processing tool, the Fourier transform, is poorly suited for analysing a signal which has abrupt transition. On the other hand, the wavelet transform has a multiresolution capability and provides information about both the position (time) and the frequency of a signal.

The wavelet transform of a signal is obtained by convolving the signal with a shifted and scaled version of the original (mother) wavelet as described by (3.43) [78].

$$Wf(s, u) = \langle f, \psi_{s,u} \rangle = \frac{1}{\sqrt{s}} \int_{-\infty}^{+\infty} f(t) \psi^* \left(\frac{t-u}{s} \right) dt \quad (3.43)$$

Where $Wf(s, u)$ denotes the continuous wavelet transform of the signal $f(t)$ for the scale parameter s and time (position) parameter u .

Therefore from a single variable function $f(t)$ we get a two-variable wavelet transform $Wf(s, u)$. Unlike the Fourier transform where the analysing function is sinusoidal, there are many different possible analysing functions (mother wavelets) for the wavelet transform and that constitutes one of its strengths as one can choose one that fits the best his application.

To qualify as mother wavelet, a function $\psi(t) \in L^2(\mathbb{R})$ should fulfil the admissibility condition (3.44) [78] where $\hat{\psi}(\omega)$ is the Fourier transform of $\psi(t)$.

$$C_{\psi} = \int_0^{\infty} \frac{|\hat{\psi}(\omega)|^2}{\omega} d\omega < +\infty \quad (3.44)$$

For the integral in (3.44) to be finite, we should have $\hat{\psi}(0) = 0$ and that is why a wavelet must have a zero average as described by (3.45).

$$\int_{-\infty}^{+\infty} \psi(t) dt = 0 \quad (3.45)$$

Figure 3-2 shows three examples of mother wavelets.

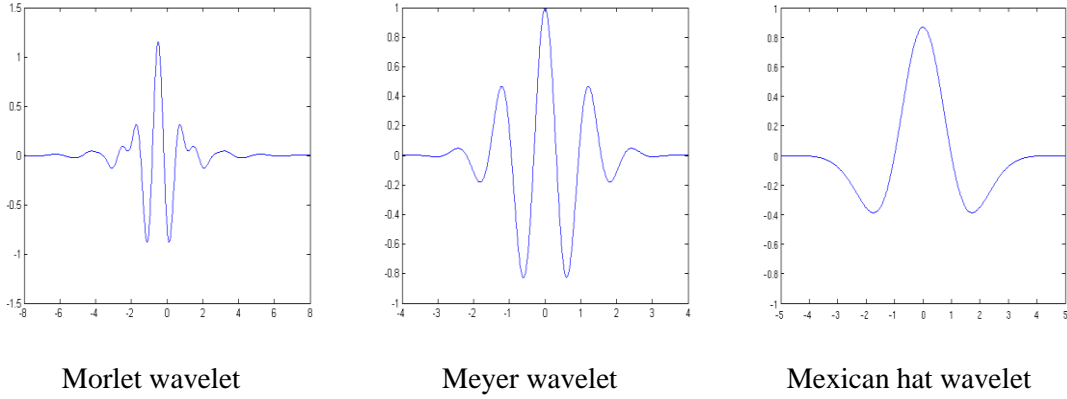


Figure 3-2: Examples of mother wavelets [124]

3.3.2 Discrete wavelet transform

The wavelet described by equation (3.43) is known as continuous wavelet transform (CWT). As specified earlier, the CWT transforms a 1D signal into a 2D signal and is therefore highly redundant.

Discrete wavelets are not continuously scalable and translatable but can only be scaled and translated in discrete steps. The discrete wavelet transform (DWT) corresponding to a CWT function $Wf(s, u)$ is obtained by sampling the coordinates (s, u) on a grid. The process, called dyadic sampling, in which the values of discrete scale as well as the corresponding sampling interval differ by a factor of two, is common. This leads to a modified wavelet representation given by (3.46)

$$\psi_{j,k}(t) = \frac{1}{\sqrt{2^j}} \left(\frac{t - k \cdot 2^j}{2^j} \right) \quad (3.46)$$

where j, k are integers.

Then the DWT of a function $f(t)$ would be given by (3.47).

$$Wf(j, k) = \langle f, \psi_{j,k} \rangle = \int_{-\infty}^{+\infty} f(t) \psi_{j,k}(t) dt \quad (3.47)$$

With a properly chosen wavelet $\psi(t)$, the family of functions $\{\psi_{j,k}(t)\}_{j,k \in \mathbb{Z}}$ can form an orthonormal basis of $L^2(\mathbb{R})$ [78]. In such cases, any finite energy signal $f(t)$ can be decomposed into a sum of dilated and translated versions of $\psi(t)$ weighted by the wavelet coefficients $Wf(j, k)$ as shown by (3.48).

$$f(t) = \sum_{k=-\infty}^{+\infty} \sum_{j=-\infty}^{+\infty} Wf(j, k) \psi_{j,k}(t) = \sum_{k=-\infty}^{+\infty} \sum_{j=-\infty}^{+\infty} \langle f, \psi_{j,k} \rangle \psi_{j,k}(t) \quad (3.48)$$

3.3.3 Multiresolution approximations

The wavelet decomposition of a signal $f(t)$ according to (3.48) allows multiresolution approximations of the signal. Specifically, the partial sum $\sum_{k=-\infty}^{+\infty} \langle f, \psi_{j,k} \rangle \psi_{j,k}(t)$ can be interpreted as the difference between two approximations of f at resolutions 2^{j+1} and 2^j . Formally, the approximation of a function at resolution 2^j is an orthogonal projection on a space $V_j \subset L^2(\mathbb{R})$, where the subspace V_j regroups all possible approximations at the resolution 2^j .

The formal definition of a multiresolution approximation is provided by Mallat [78] as follows: A sequence $\{V_j\}_{j \in \mathbb{Z}}$ of closed subspaces of $L^2(\mathbb{R})$ is a multiresolution approximation if the properties (3.49) through (3.54) are satisfied.

$$\forall (j, k) \in \mathbb{Z}, f(t) \in V_j \Leftrightarrow f(t - 2^j k) \in V_j \quad (3.49)$$

$$\forall j \in \mathbb{Z}, V_{j+1} \subset V_j \quad (3.50)$$

$$\forall j \in \mathbb{Z}, f(t) \in V_j \Leftrightarrow f\left(\frac{t}{2}\right) \in V_{j+1} \quad (3.51)$$

$$\lim_{j \rightarrow +\infty} V_j = \bigcap_{j=-\infty}^{+\infty} V_j = \{0\} \quad (3.52)$$

$$\lim_{j \rightarrow -\infty} V_j = \text{closure} \left(\bigcup_{j=-\infty}^{+\infty} V_j \right) = L^2(\mathbb{R}) \quad (3.53)$$

$$\text{There exists } \theta \text{ such that } \{\theta(t - n)\}_{n \in \mathbb{Z}} \text{ is a Riesz basis of } V_0 \quad (3.54)$$

Once the properties (3.49) through (3.54) are satisfied, it has been proven [78] that there exists a unique function $\phi(t) \in L^2(\mathbb{R})$, called a **scaling function**, so that the family $\left\{ \phi_{j,k}(t) = \frac{1}{\sqrt{2^j}} \phi\left(\frac{t-2^j k}{2^j}\right) \right\}_{k \in \mathbb{Z}}$ is an orthogonal basis of V_j for all $j \in \mathbb{Z}$.

Given such function then the approximation of a function $f(t) \in L^2(\mathbb{R})$ at resolution 2^j can be expressed by (3.55).

$$P_{V_j} f = \sum_{k=-\infty}^{+\infty} a_j[k] \phi_{j,k}(t) \quad (3.55)$$

Where the coefficients $a_j[k]$ provide the discrete approximation of f at resolution 2^j and are given by (3.56)

$$a_j[k] = \langle f, \phi_{j,k} \rangle = \int_{-\infty}^{+\infty} f(t) \frac{1}{\sqrt{2^j}} \phi\left(\frac{t - 2^j k}{2^j}\right) dt = f * \bar{\phi}(2^j k) \quad (3.56)$$

Where $\bar{\phi}_j(t) = \frac{1}{\sqrt{2^j}} \phi\left(\frac{-t}{2^j}\right)$ and $*$ denotes the convolution operator.

Property (3.51) implies that $V_1 \subset V_0$. In particular since $\frac{1}{\sqrt{2}} \phi\left(\frac{t}{2}\right) \in V_1, \frac{1}{\sqrt{2}} \phi\left(\frac{t}{2}\right) \in V_0$. Since $\{\phi_{0,k}(t) = \phi(t - k)\}_{k \in \mathbb{Z}}$ is an orthonormal basis of V_0 we can decompose $\frac{1}{\sqrt{2}} \phi\left(\frac{t}{2}\right)$ as shown by (3.57),

$$\frac{1}{\sqrt{2}} \phi\left(\frac{t}{2}\right) = \sum_{k=-\infty}^{+\infty} h_0[k] \phi(t - k) \quad (3.57)$$

with $h_0[k]$ given by (3.58).

$$h_0[k] = \left\langle \frac{1}{\sqrt{2}} \phi\left(\frac{t}{2}\right), \phi(t - k) \right\rangle \quad (3.58)$$

The sequence $h_0[k]$ is interpreted as a discrete filter.

The Fourier transform of (3.57) yields (3.59)

$$\hat{\phi}(2\omega) = \frac{1}{\sqrt{2}} \hat{h}_0(\omega) \hat{\phi}(\omega) \quad (3.59)$$

From (3.59) we get (3.60) and then (3.61)

$$\hat{\phi}(\omega) = \frac{1}{\sqrt{2}} \hat{h}_0\left(\frac{\omega}{2}\right) \hat{\phi}\left(\frac{\omega}{2}\right) \quad (3.60)$$

$$\hat{\phi}(\omega) = \prod_{p=1}^{+\infty} \frac{\hat{h}_0(2^{-p} \omega)}{\sqrt{2}} \hat{\phi}(0) \quad (3.61)$$

Mallat [78] proved that for the infinite product in (3.61) to converge to the Fourier transform of a scaling function, the Fourier series of $h_0[k]$ must satisfy (3.62) and (3.63).

$$\forall \omega \in \mathbb{R}, \left| \hat{h}_0(\omega) \right|^2 + \left| \hat{h}_0(\omega + \pi) \right|^2 = 2 \quad (3.62)$$

$$\hat{h}_0(0) = \sqrt{2} \quad (3.63)$$

Discrete filters that satisfy (3.62) are called conjugate mirror filters. Therefore, from a conjugate mirror filter that satisfies (3.63), one can construct a scaling function by using (3.61). In Subsection 3.3.4, we will see that from the scaling function one gets a wavelet.

3.3.4 Wavelet representations

From a scaling function $\phi(t)$ and its corresponding conjugate mirror filter $h_0[k]$ one can construct a wavelet function $\psi(t)$ whose Fourier transform $\hat{\psi}(\omega)$ satisfies (3.64)

$$\hat{\psi}(\omega) = \frac{1}{\sqrt{2}} \hat{h}_1\left(\frac{\omega}{2}\right) \hat{\phi}\left(\frac{\omega}{2}\right) \quad (3.64)$$

with function $\hat{h}_1(\omega)$ given by (3.65)

$$\hat{h}_1(\omega) = e^{-j\omega} \hat{h}_0^*(\omega + \pi) \quad (3.65)$$

Let \mathbf{W}_j denote the orthogonal complement of \mathbf{V}_j on \mathbf{V}_{j+1} . The detail signal of $f(t)$ at resolution 2^{-j} can be obtained by orthogonal projection of $f(t)$ onto \mathbf{W}_j . The family $\left\{\psi_{j,k}(t) = \frac{1}{\sqrt{2^j}} \psi\left(\frac{t-2^j k}{2^j}\right)\right\}_{k \in \mathbb{Z}}$ where the Fourier transform of $\psi(t)$ is given by (3.64) is an orthogonal basis of \mathbf{W}_j for all $j \in \mathbb{Z}$ and allows us to calculate that projection. Based on the orthogonal basis $\{\psi_{j,k}(t)\}_{k \in \mathbb{Z}}$ of \mathbf{W}_j , the detail signal of $f(t)$ at resolution 2^{-j} can be expressed by (3.66)

$$P_{\mathbf{W}_j} f = \sum_{k=-\infty}^{+\infty} d_j[k] \psi_{j,k}(t) \quad (3.66)$$

where the coefficients $d_j[k]$ characterise the detail signal of f at resolution 2^{-j} and are given by (3.67)

$$d_j[k] = \langle f, \psi_{j,k} \rangle = \int_{-\infty}^{+\infty} f(t) \frac{1}{\sqrt{2^j}} \psi\left(\frac{t-2^j k}{2^j}\right) dt = f * \bar{\psi}(2^j k) \quad (3.67)$$

where $\bar{\psi}_j(t) = \frac{1}{\sqrt{2^j}} \psi\left(\frac{-t}{2^j}\right)$ and $*$ denotes the convolution operator.

Based on the multiresolution approximations and the difference of information between the approximations at two successive resolutions, a signal $f(t) \in L^2(\mathbb{R})$ can be described by orthogonal wavelet representations given by (3.68).

$$\begin{aligned}
 f(t) &= P_{V_j} f + \sum_{i=-\infty}^j P_{W_i} f = \sum_{k=-\infty}^{+\infty} a_j[k] \phi_{j,k}(t) + \sum_{i=-\infty}^j \sum_{k=-\infty}^{+\infty} d_i[k] \psi_{i,k}(t) \\
 &= \sum_{k=-\infty}^{+\infty} \langle f, \phi_{j,k} \rangle \phi_{j,k}(t) + \sum_{i=-\infty}^j \sum_{k=-\infty}^{+\infty} \langle f, \psi_{i,k} \rangle \psi_{i,k}(t)
 \end{aligned} \tag{3.68}$$

The first term of (3.68) represents the approximation of the signal f at resolution 2^j , while the second term represents the detail signals at resolutions 2^i , $i \leq j$.

3.3.5 FWT algorithm

In many practical applications the signal of interest is sampled. The DWT of a sampled signal $f[n]$ is given by (3.69)

$$Wf(j, k) = \langle \psi_{j,k}, f \rangle = \sum_{n=-\infty}^{+\infty} \psi_{j,k}[n] \cdot f[n] \tag{3.69}$$

The DWT transform of a sampled signal can be more easily computed using the fast wavelet transform (FWT) developed by Mallat [125]. This algorithm is a classical scheme known as a two-channel filter bank coder using conjugate mirror filters. The single-level decomposition can be described by Figure 3-3.

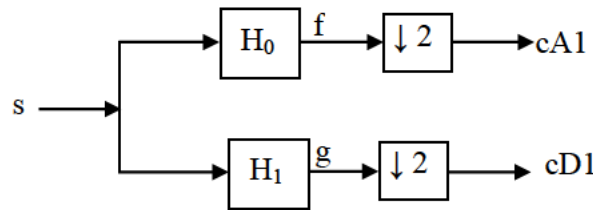


Figure 3-3: Single-level DWT decomposition

The original sampled signal s is convolved with the low-pass filter H_0 and the result f is downsampled (taking one sample out of two) to produce the first level wavelet approximation coefficients $cA1$, which has about half as many samples as the original signal. Similarly the original sampled signal s is convolved with the high-pass filter H_1 and the result g is downsampled to produce the first level wavelet detail coefficients $cD1$. The number of samples of $cD1$ is about half the number of samples of the original signal s . A reverse process can be used to reconstruct the original signal from the wavelet approximation and detail coefficients as illustrated in Figure 3-4. (H_0, H_1) and (F_0, F_1) form a system of specially designed filters called **quadrature mirror filters**.

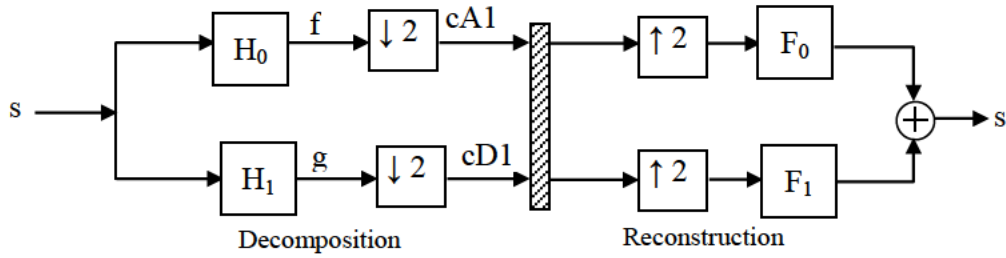


Figure 3-4: Single-level DWT decomposition and reconstruction

For a multilevel decomposition, the approximation coefficients from a previous level is decomposed using the process described above to give the next level wavelet approximation coefficients and the next level wavelet details coefficients. Figure 3-5 shows a three-level wavelet decomposition.

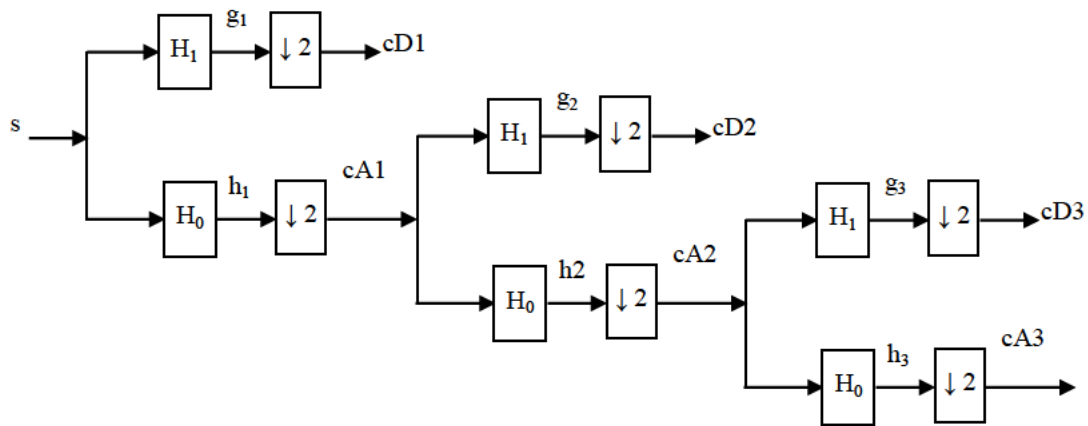


Figure 3-5: Three-level DWT decomposition

3.3.6 The undecimated discrete wavelet transform

The DWT as described previously by the FWT algorithm is common but in some applications it is inappropriate because it is not shift invariant. In such cases the undecimated discrete wavelet transform (UDWT) can be used. Single-level UDWT is similar to single-level DWT with the exception that the results of filtering the original signal are not downsampled. Therefore, one ends up with the wavelet approximation coefficients with as many samples as the original signal and the wavelet detail coefficients with as many coefficients as the original signal leading to the overall results with twice as much data as the original signal. Figure 3-6 illustrates single-level UDWT decomposition and reconstruction.

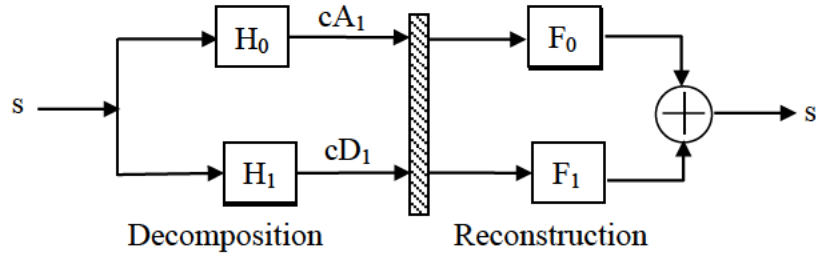


Figure 3-6: Single-level UDWT decomposition and reconstruction

For multilevel UDWT the filters coefficients of H_0 , H_1 , F_0 and F_1 are upsampled from one level to the next one, i.e. coefficients of value 0 are introduced between existing coefficients. Figure 3-7 illustrates a two-level UDWT decomposition and reconstruction.

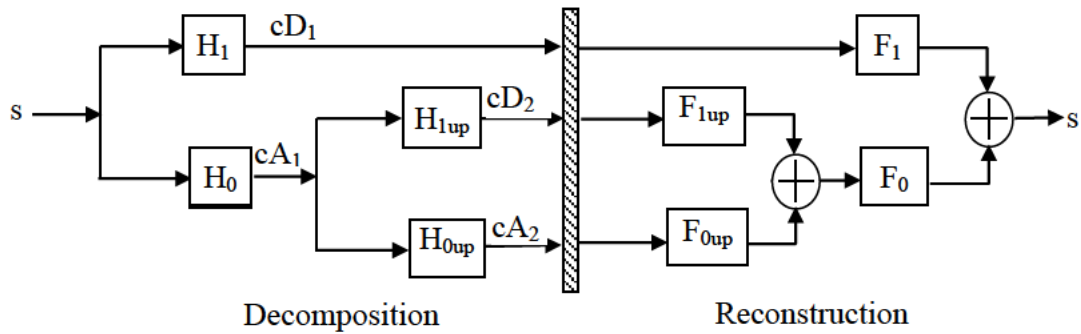


Figure 3-7: Two-level UDWT decomposition and reconstruction.

H_{0up} , H_{1up} , F_{0up} , F_{1up} are upsampled versions of H_0 , H_1 , F_0 and F_1 respectively

3.3.7 Two-dimensional wavelet transform

Wavelet transform can be applied on two-dimensional signals (images) using the FWT algorithm in two phases. In phase one, the filtering (high-pass and low-pass), followed by downsampling, is done along the columns. In phase two, the filtering (high-pass and low-pass), followed by downsampling, is applied to the results of phase one along the rows. Figure 3-8 illustrates this process.

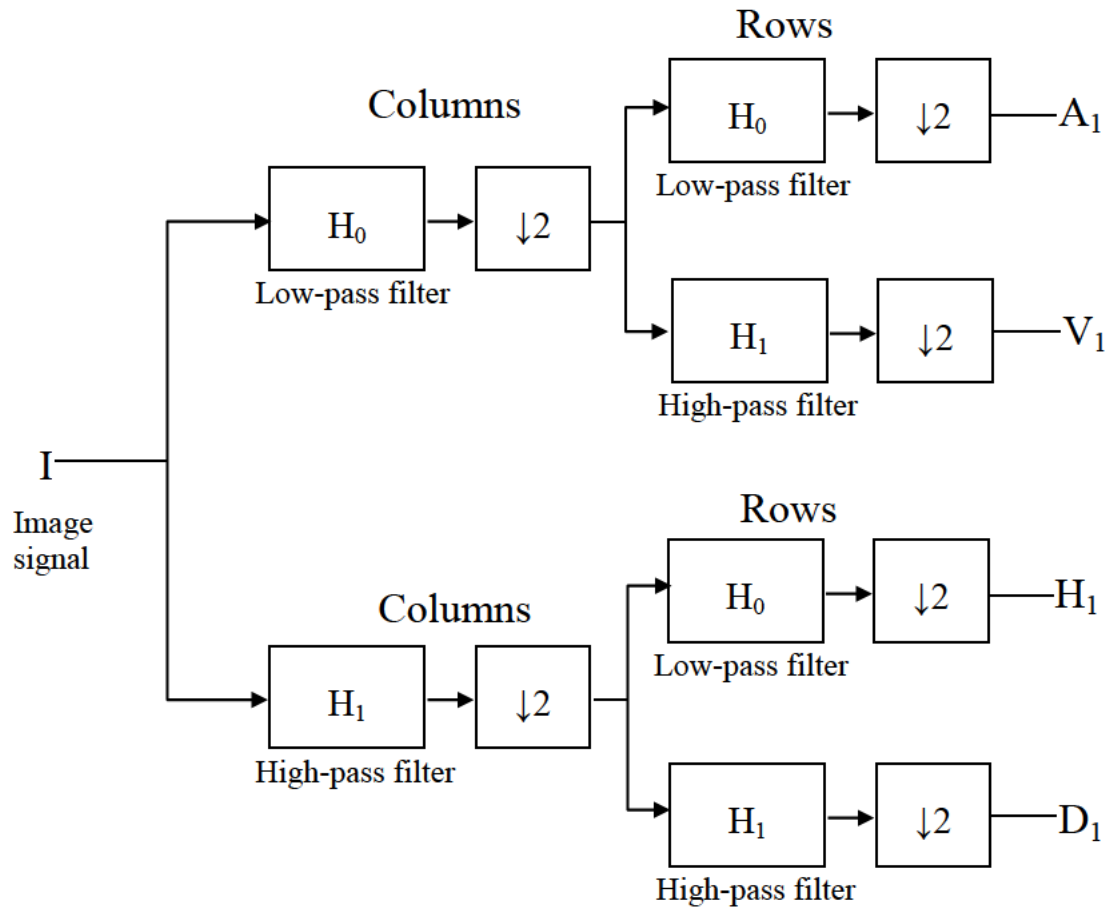


Figure 3-8: One level wavelet decomposition of a 2D signal

The results are the approximation sub-image A and three detail sub-images: vertical sub-image V_1 (with vertical edges but no horizontal edges), horizontal sub-image H_1 (with horizontal edges but no vertical edges) and diagonal sub-image V_1 (with no horizontal nor vertical edges). The size of each sub-image is a quarter of the original image, as illustrated by Figure 3-9 and Figure 3-10.

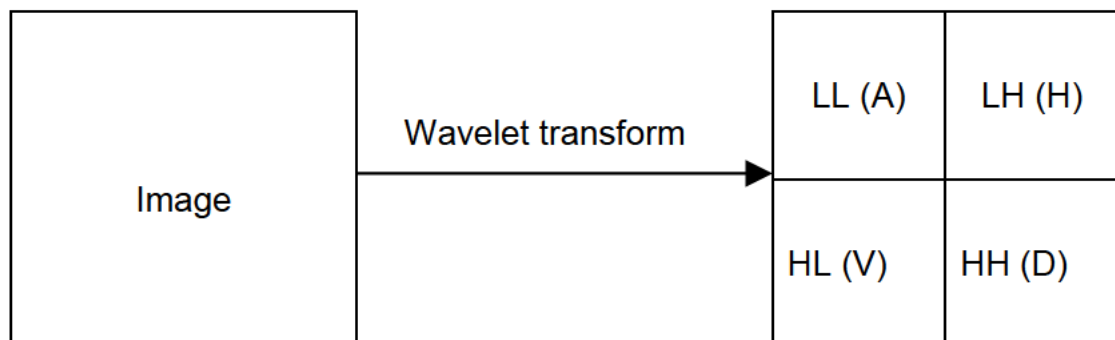


Figure 3-9: Wavelet decomposition of an image

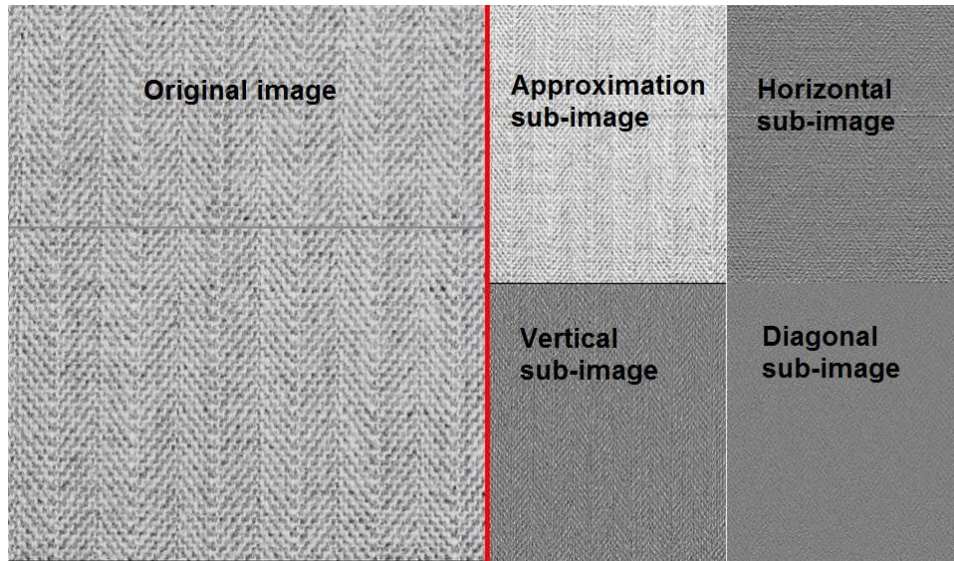


Figure 3-10: Example of wavelet transform of an image

For multilevel decomposition, the approximation sub-image is further decomposed into approximation sub-image and horizontal, vertical and diagonal sub-images of the next level.

One of the main differences between textile inspection and steel or paper inspection is that the textile image is rich in details when compared to steel or paper images. That is the reason why textile inspection requires attenuating the texture background and accentuating defects [25]. Wavelet transforms can be used to that effect.

3.3.8 The dual-tree complex wavelet transform

The discrete wavelet transform (DWT) and multiscale analysis as presented in subsections 3.3.2 and 3.3.3 have proven to be useful in a wide range of applications. The primary reason is because they provide an extremely efficient representation for many types of signals that appear often in practice but are not well represented by the Fourier analysis [72].

However, DWT suffers from two main drawbacks that makes it less suitable for applications such as pattern recognition: (i) it is not shift-invariant, meaning the same pattern in a signal or image may lead to different wavelet coefficients depending on its location in the signal or image, and (ii) its directional selectivity in two and higher dimensions is low. The latter setback refers, for instance, to the fact that DWT cannot distinguish features in an image that appear parallel to the main diagonal from those that appear parallel to the secondary diagonal.

The dual-tree complex wavelet transform was proposed as a solution to the above-mentioned problems of DWT [99]. It represents a signal $f(t)$ in a way similar to DWT as shown by (3.70)

$$f(t) = \sum_{k=-\infty}^{+\infty} c_c(k) \phi_c(t-k) + \sum_{j=0}^{+\infty} \sum_{k=-\infty}^{+\infty} d_c(j,k) 2^{j/2} \psi_c(2^j t - k) \quad (3.70)$$

where the wavelet approximation $c_c(k)$ and details $d_c(j,k)$ coefficients are computed by inner products shown in (3.71) and (3.72).

$$c_c(k) = \langle f(t), \phi_c(t-k) \rangle = \int_{-\infty}^{+\infty} f(t) \phi_c(t-k) dt \quad (3.71)$$

$$d_c(j,k) = \langle f(t), 2^{j/2} \psi_c(2^j t - k) \rangle = 2^{j/2} \int_{-\infty}^{+\infty} f(t) \psi_c(2^j t - k) dt \quad (3.72)$$

The difference with DWT is that the scaling function $\phi_c(t)$ and wavelet $\psi_c(t)$ are complex rather than real and can be written as (3.73) and (3.74).

$$\phi_c(t) = \phi_h(t) + j \phi_g(t) \quad (3.73)$$

$$\psi_c(t) = \psi_h(t) + j \psi_g(t) \quad (3.74)$$

The real $\phi_h(t)$ and imaginary $\phi_g(t)$ parts of the scaling function, and the real $\psi_h(t)$ and imaginary $\psi_g(t)$ parts of the wavelet are designed so that they form approximate Hilbert pairs [72]. That means that the scaling function $\phi_c(t)$ and wavelet $\psi_c(t)$ are approximately analytic, that their Fourier transforms $\hat{\phi}_c(\omega)$ and $\hat{\psi}_c(\omega)$ are approximately single sided as described by (3.75) and (3.76).

$$\hat{\phi}_c(\omega) \approx 0, \quad \text{for } \omega < 0 \quad (3.75)$$

$$\hat{\psi}_c(\omega) \approx 0, \quad \text{for } \omega < 0 \quad (3.76)$$

Practically, the decomposition (3.70) is implemented using two real filter bank trees proposed by Kingsbury [99-100], hence the name dual-tree complex wavelet transform. Figure 3-11 illustrates such decomposition. The coefficients obtained from tree-h and tree-g are interpreted respectively as real and imaginary parts of the transform.

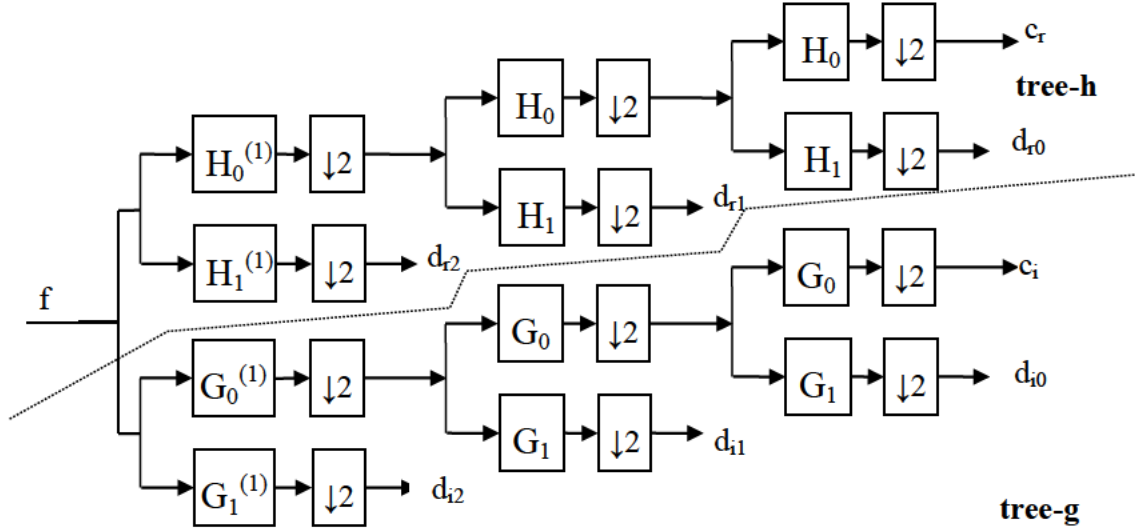


Figure 3-11: DTCWT decomposition of a signal using two real filter bank trees

In Figure 3-11, H_0 and H_1 are respectively the low- and high-pass decomposition filters for tree-h, while G_0 and G_1 are respectively the low- and high-pass decomposition filters for tree-g. The first level filters ($H_0^{(1)}$, $H_1^{(1)}$) and ($G_0^{(1)}$, $G_1^{(1)}$) can be any set of orthogonal or biorthogonal wavelet filters [72].

The extension to two dimensions is achieved by using six 2D complex wavelets given by (3.77) through (3.79) and two 2D scaling functions described by (3.80), where $\psi(x)$ is a complex (approximately analytic) wavelet given by $\psi(x) = \psi_h(x) + j\psi_g(x)$, $\phi(x)$ is a complex (approximately analytic) scaling function given by $\phi(x) = \phi_h(x) + j\phi_g(x)$ and $*$ denotes the complex conjugation operator.

$$\psi_{1,1}(x, y) = \phi(x)\psi(y), \quad \psi_{1,2}(x, y) = \phi(x)\psi(y)^* \quad (3.77)$$

$$\psi_{2,1}(x, y) = \psi(x)\phi(y), \quad \psi_{2,2}(x, y) = \psi(x)\phi(y)^* \quad (3.78)$$

$$\psi_{3,1}(x, y) = \psi(x)\psi(y), \quad \psi_{3,2}(x, y) = \psi(x)\psi(y)^* \quad (3.79)$$

$$\phi_1(x, y) = \phi(x)\phi(y), \quad \phi_2(x, y) = \phi(x)\phi(y)^* \quad (3.80)$$

Developing (3.77) through (3.80) gives (3.81) through (3.88).

$$\psi_{1,1}(x, y) = (\phi_h(x)\psi_h(y) - \phi_g(x)\psi_g(y)) + j(\phi_g(x)\psi_h(y) + \phi_h(x)\psi_g(y)) \quad (3.81)$$

$$\psi_{1,2}(x, y) = (\phi_h(x)\psi_h(y) + \phi_g(x)\psi_g(y)) + j(\phi_g(x)\psi_h(y) - \phi_h(x)\psi_g(y)) \quad (3.82)$$

$$\psi_{2,1}(x, y) = (\psi_h(x)\phi_h(y) - \psi_g(x)\phi_g(y)) + j(\psi_g(x)\phi_h(y) + \psi_h(x)\phi_g(y)) \quad (3.83)$$

$$\psi_{2,2}(x, y) = (\psi_h(x)\phi_h(y) + \psi_g(x)\phi_g(y)) + j(\psi_g(x)\phi_h(y) - \psi_h(x)\phi_g(y)) \quad (3.84)$$

$$\psi_{3,1}(x, y) = (\psi_h(x)\psi_h(y) - \psi_g(x)\psi_g(y)) + j(\psi_g(x)\psi_h(y) + \psi_h(x)\psi_g(y)) \quad (3.85)$$

$$\psi_{3,2}(x, y) = (\psi_h(x)\psi_h(y) + \psi_g(x)\psi_g(y)) + j(\psi_g(x)\psi_h(y) - \psi_h(x)\psi_g(y)) \quad (3.86)$$

$$\phi_1(x, y) = (\phi_h(x)\phi_h(y) - \phi_g(x)\phi_g(y)) + j(\phi_g(x)\phi_h(y) + \phi_h(x)\phi_g(y)) \quad (3.87)$$

$$\phi_2(x, y) = (\phi_h(x)\phi_h(y) + \phi_g(x)\phi_g(y)) + j(\phi_g(x)\phi_h(y) - \phi_h(x)\phi_g(y)) \quad (3.88)$$

The implementation of 2D DTCWT consists of two steps; [126]

- (i) An input image is decomposed up to a desired level by two separable 2D DWT branches, with filters H_0, H_1, G_0, G_1 designed to meet the approximate Hilbert pair requirement. Then six high-pass subbands are generated at each level.
- (ii) Every two corresponding subbands which have the same passbands are combined by averaging or differencing.

3.3.9 The wavelet transform features

Once the wavelet coefficients are computed using an appropriate wavelet transform algorithm, some statistics, referred to as features, are computed from the coefficients of each channel to characterise the texture of the image.

The most commonly used feature is the channel variance, which is the mean energy of wavelet coefficients of that channel as described by (3.89) [12, 16, 88]

$$f^k = \frac{1}{MN} \sum_{i=1}^M \sum_{j=1}^N (d_{i,j}^k)^2 \quad (3.89)$$

where f^k is the variance for the k^{th} channel and $d_{i,j}^k$ is $(i, j)^{\text{th}}$ wavelet coefficient of k^{th} channel.

It is to be realised that the channel variance is also the mean energy of wavelet coefficients because the sum of (detail) wavelet coefficients and therefore their mean is always zero [127].

The mean absolute values of wavelet coefficients of wavelet channel, described by (3.90), are also used [96, 128].

$$f^k = \frac{1}{MN} \sum_{i=1}^M \sum_{j=1}^N |d_{i,j}^k| \quad (3.90)$$

Alternatively, the sum of squares or sum of absolute values of wavelet coefficients are also used [98, 129, 130].

Some authors combined the wavelet methods with second order statistical methods. They calculated GLCM features [21, 23, 130, 131] or features based on the grey level difference method (GLDM) [132] from wavelet coefficients. Others proposed entropy of magnitudes of wavelet coefficients [133], the principal components of wavelet coefficients [134] and a subset of the wavelet coefficients with the highest magnitude [135] as features for texture characterisation.

For complex wavelet coefficients obtained using DTCWT, some of the features used are the means and standard deviations of the complex wavelet coefficients [102, 105, 136], the variance and entropy of magnitudes and relative phases of complex wavelet coefficients [137] and the Frobenius norm of magnitude of wavelet coefficients [101]. The Frobenius norm of a matrix is defined as the square root of the sum of the absolute squares of its elements as described by (3.91).

$$\|S\|_{Fro} = \sqrt{\sum_{i=1}^M \sum_{j=1}^N |s_{i,j}|^2} \quad \text{where } S \text{ is a } M \times N \text{ matrix} \quad (3.91)$$

3.4 Markov random field model-based features

3.4.1 What is a Markov random field?

Let X represent a greyscale textured image. In the Markov random field model the greyscale value of each pixel as a random variable. By considering all the pixels, the corresponding random variables constitute together a random field. A Markov random field (MRF) is a random field that possesses the Markovian property – the value of a pixel depends directly only on the values of neighbouring pixels and on no other pixels [138]. That indicates that only neighbouring pixels have direct interaction with each other.

3.4.2 Neighbourhood systems

The sites (pixels in the case of an image) in a lattice S are related to one another via a neighbourhood system. A neighbourhood system for S is defined by (3.92) [109]

$$N = \{N_i \mid \forall i \in S\} \quad (3.92)$$

where N_i is the set of sites neighbouring i . The neighbouring relationship has the following properties:

- (i) A site is not neighbouring to itself: $i \notin N_i$
- (ii) The neighbouring relationship is mutual: $i' \in N_i \Leftrightarrow i \in N_{i'}$

For a regular lattice S , the set of neighbours of i is defined as a set of sites within a radius of \sqrt{r} from i as given by (3.93).

$$N_i = \{i' \in S \mid [(\text{dist}(\text{pixel } i, \text{pixel } i'))^2 \leq r, i' \neq i]\} \quad (3.93)$$

In a first order neighbourhood system every (interior) site has four neighbours, while in a second order neighbourhood system there are eight neighbours for every (interior) site. Figure 3-12 shows neighbourhood systems up to the fifth order. The number $n=1, 2, \dots, 5$ indicates the outermost neighbouring sites in the n^{th} order neighbourhood system.

5	4	3	4	5
4	2	1	2	4
3	1	X	1	3
4	2	1	2	4
5	4	3	4	5

Figure 3-12: Neighbourhood systems up to the fifth order

A clique c for (N, S) is defined as a subset of sites in S . It consists of either a single site $c=\{i\}$, a pair of neighbouring sites $c=\{i, i'\}$, a triple of neighbouring sites $c=\{i, i', i''\}$, and so on. Figure 3-13 shows cliques for a second order neighbourhood system.

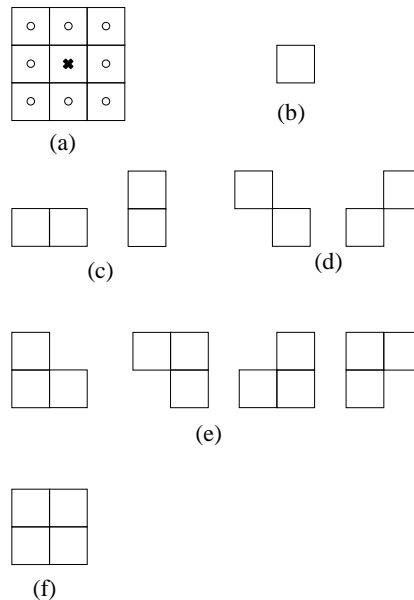


Figure 3-13: Cliques for a second order neighbourhood system

- (a) Neighbourhood system (b) Single site (c) Horizontal and vertical pair-site cliques (d) Diagonal pair-site cliques (e) Triple-site cliques (f) Quadriple-site clique

3.4.3 Markov random field models

For a given neighbourhood system, the Markovian property of a Markov random field can be expressed as (3.94).

$$P(f_i | f_{S-\{i\}}) = P(f_i | f_{N_i}) \quad (3.94)$$

There are two approaches to specify an MRF: (i) in terms of conditional probabilities $P(f_i | f_{N_i})$ and (ii) in terms of the joint probability $P(f)$. A theorem by Hammersley-Clifford [108] establishes the equivalence between the two approaches. It states that F is an MRF on a lattice S with respect to a neighbourhood system N if and only if F is a Gibbs random field on S with respect to N .

A set of random variables F is said to be a Gibbs random field on S with respect to N if and only if its configurations obey a Gibbs distribution [109]. A Gibbs distribution takes the form (3.95)

$$P(f) = \frac{1}{Z} e^{-\frac{1}{T} U(f)} \quad (3.95)$$

where Z , given by (3.96), is a normalising constant called the partition function. T is a constant called temperature (normally assumed to be 1) and $U(f)$ is the energy function.

$$Z = \sum_{f \in F} e^{-\frac{1}{T} U(f)} \quad (3.96)$$

$$U(f) = \sum_{c \in C} V_c(f) \quad (3.97)$$

The energy function $U(f)$ given by (3.97) is the sum of all clique potentials over all possible cliques C .

When we consider cliques potentials of up to two sites, the energy takes the form given by (3.98).

$$U(f) = \sum_{i \in S} V_1(f_i) + \sum_{i \in S} \sum_{i' \in N_i} V_2(f_i, f_{i'}) \quad (3.98)$$

When $V_1(f_i) = f_i G_i(f_i)$ and $V_2(f_i, f_{i'}) = \beta_{i,i'} f_i f_{i'}$, where $G_i(\cdot)$ are arbitrary functions and $\beta_{i,i'}$ are constants reflecting the pair-site interaction between i and i' , the energy is given by (3.99).

$$U(f) = \sum_{\{i\} \in C_1} f_i G_i(f_i) + \sum_{\{i,i'\} \in C_2} \beta_{i,i'} f_i f_{i'} \quad (3.99)$$

Such models are called auto-models [109].

An auto-model is said to be **auto-logistic** if the f_i 's take on values in the discrete label set $\{0, 1\}$ or $\{-1, +1\}$. The corresponding energy is of the form (3.100).

$$U(f) = \sum_{\{i\} \in C_1} \alpha_i f_i + \sum_{\{i,i'\} \in C_2} \beta_{i,i'} f_i f_{i'} \quad (3.100)$$

An auto-model is said to be an **auto-binomial** model if the f_i 's take on values in $\{0, 1, \dots, M-1\}$ and every f_i has a conditional binomial distribution of M trials and probability of success q as given by (3.101)

$$P(f_i | f_{N_i}) = \binom{M-1}{f_i} q^{f_i} (1-q)^{M-1-f_i} \quad (3.101)$$

$$q = \frac{e^{\alpha_i + \sum_{i' \in N_i} \beta_{i,i'} f_{i'}}}{1 + e^{\alpha_i + \sum_{i' \in N_i} \beta_{i,i'} f_{i'}}}$$

where

The corresponding energy takes the form (3.102).

$$U(f) = - \sum_{\{i\} \in C_1} \ln \binom{M-1}{f_i} - \sum_{\{i\} \in C_1} \alpha_i f_i + \sum_{\{i,i'\} \in C_2} \beta_{i,i'} f_i f_{i'} \quad (3.102)$$

An auto-model is said to be **auto-normal** model, also called Gaussian MRF, if the label set is continuous (real line) and the joint distribution is multivariate normal. Its conditional density function is given by (3.103).

$$p(f_i | f_{N_i}) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp \left(-\frac{1}{2\sigma^2} \left[f_i - \mu_i - \sum_{i' \in N_i} \beta_{i,i'} (f_{i'} - \mu_{i'}) \right]^2 \right) \quad (3.103)$$

It is the normal distribution with conditional mean given by (3.104) and conditional variance given by (3.105), where $\beta_{i,i'}$ are the model parameters.

$$E(f_i | f_{N_i}) = \mu_i - \sum_{i' \in N_i} \beta_{i,i'} (f_{i'} - \mu_{i'}) \quad (3.104)$$

$$\text{var}(f_i | f_{N_i}) = \sigma^2 \quad (3.105)$$

An auto-logistic model can be generalised to a **multilevel logistic** (MLL), also called a Strauss process or generalised Ising model [107]. There are M discrete labels in the label set $\{1, 2, \dots, M\}$ and in this type of model a clique potential depends on type c (related to size, shape and possibly orientation) and on the local configuration. For cliques containing more than one site, the MLL clique potentials are defined by (3.106)

$$V_c(f) = \begin{cases} \zeta_c & \text{if all sites on } c \text{ have the same label} \\ -\zeta_c & \text{otherwise} \end{cases} \quad (3.106)$$

where ζ_c is the potential for type c cliques. For single site cliques, the MLL clique potentials depend on the label assigned to the site (grey value of a pixel for images) as given by (3.107)

$$V_c(f) = V_c(f_i) = \alpha_I \text{ if } f_i = I \quad (3.107)$$

where α_I is the potential for label value I .

3.4.4 MRF texture modelling

To model a texture as a Markov random field, one has first to choose the type of neighbourhood, and then choose a specific Markov model characterised by the local conditional probability function or by the energy function containing Gibbs clique potentials [138].

The Gaussian Markov random fields (GMRFs) are commonly used for modelling textures. For example, Zhao et al. used them to classify high resolution spatial satellite imagery [139]. Chellappa and Chatterjee [111] used GMRF models to classify textures from the Brodatz album. Torres and Juan [140] compared the discriminating ability of GMRF features and features from Gabor convolution energies to classify land cover types in Synthetic Aperture Radar (SAR) imagery. Cohen et al. [113] used GMRF to detect and locate the various kinds of defects that might be present in a given fabric sample based on its image. In chapter 7 we will evaluate the performance of GMRF features for the purpose of fabric defect detection and classification.

A GMRF model assumes that the local conditional probability that specifies the spatial dependencies of a pixel and its neighbouring pixels is Gaussian as given by (3.103). On the other hand, the grey value of the central pixel can be represented as a linear combination of values of neighbouring pixels and an additive noise [141] as given by (3.108)

$$f_i - \mu_i = \sum_{i' \in N_i} \beta_{i,i'} (f_{i'} - \mu_{i'}) + e_i \quad (3.108)$$

where e_i is a Gaussian noise sequence with zero mean and autocorrelation function given by (3.109).

$$R_n(i, i') = \begin{cases} \sigma^2 & \text{if } i = i' \\ -\sigma^2 \beta_{i, i'} & \text{if } i' \in N_i \\ 0 & \text{otherwise} \end{cases} \quad (3.109)$$

The parameters $\beta_{i, i'}$ and the conditional variance σ^2 describe the GMRF model and characterise the texture.

3.4.5 MRF texture features and feature extraction

In MRF modelling, a texture is assumed to be a realisation of a MRF and to model a texture is to specify the corresponding conditional probabilities of Gibbs clique potentials. Texture features are then the MRF model parameters and feature extraction is equivalent to parameter estimation.

Estimation of MRF model parameters given its realisation is difficult because the partition function Z depends on the parameters and yet the partition function is computationally intractable – it cannot be computed. Approximate parameters estimation can be done using methods such as (i) coding, (ii) maximum pseudo-likelihood and (iii) least squares parameter estimation (LSE) [109].

Using the coding method, the lattice S is divided into several disjoint set $S^{(k)}$, called codings, so that no two sites in one $S^{(k)}$ are neighbours. Under the Markovian assumption, the variable associated with the sites in an $S^{(k)}$, given the labels of all the other sites, are mutually independent. Therefore the likelihood for each coding is given by (3.110), where θ represents the model parameters.

$$P^{(k)}(f | \theta) = \prod_{i \in S^{(k)}} P(f_i | f_{N_i}, \theta) \quad (3.110)$$

Maximising (3.110) gives coding estimates $\hat{\theta}^{(k)}$. Then the arithmetic average (3.111) is usually used to get the estimate of the model parameters $\hat{\theta}$.

$$\hat{\theta} = \frac{1}{K} \sum_k \arg \max_{\theta} P^{(k)}(f | \theta) \quad (3.111)$$

For the maximum pseudo-likelihood method, the likelihood function given by (3.112) is approximated by the pseudo-likelihood defined by (3.113), where ∂S denotes the set of points at the boundaries of S .

$$P(f | \theta) = \frac{1}{Z(\theta)} e^{-U(f|\theta)} \quad (3.112)$$

$$PL(f | \theta) = \prod_{i \in S - \partial S} P(f_i | f_{N_i}, \theta) = \prod_{i \in S - \partial S} \frac{e^{-U_i(f_i | f_{N_i}, \theta)}}{\sum_{f_i} e^{-U_i(f_i | f_{N_i}, \theta)}} \quad (3.113)$$

The pseudo-likelihood given by (3.113) does not involve the partition function $Z(\theta)$ and therefore can be computed. The model parameters can be estimated by maximising the pseudo-likelihood as (3.114), or by equivalently maximising the log-likelihood as (3.115).

$$\hat{\theta} = \arg \max_{\theta} (PL(f | \theta)) \quad (3.114)$$

$$\hat{\theta} = \arg \max_{\theta} \ln (PL(f | \theta)) \quad (3.115)$$

The LSE method is easy to implement and, in the case the GMRF model, the estimate $\hat{\beta}$ of the parameters $\beta_{i,i'}$ is given by (3.116), while the estimate $\hat{\sigma}^2$ of the conditional variance is given by (3.117) [139]

$$\hat{\beta} = \left(\sum_{i \in S} Q(i) Q^T(i) \right)^{-1} \left(\sum_{i \in S} Q(i) (f_i - \mu_i) \right) \quad (3.116)$$

$$\hat{\sigma}^2 = \frac{1}{MN} \left(\sum_{i \in S} (f_i - \mu_i) Q^T(i) \hat{\beta} \right) \quad (3.117)$$

where $Q(i)$ is the column matrix whose elements are the sum of $f_i' + f_i$; the sum of pixel values diagonally opposed with respect to the central pixel.

Speis and Healey studied the LSE method and concluded that it is consistent, more efficient than coding and computationally advantageous [142].

3.5 Classifiers

3.5.1 Pattern recognition basic concepts

Classifiers belong to the field of pattern recognition and classification. The underlying problem can be stated as follows. Given a number of predefined classes (or categories), a new object (or pattern) is to be assigned to one of them. Usually the classification decision will be based on the existence of some representative sample patterns of each of the classes, called training samples (or training patterns).

A classifier is an algorithm that uses some strategy to assign a new pattern to a class based on the knowledge of the training patterns. One such algorithm is the nearest-neighbour algorithm. Davies [143] states that the nearest-neighbour algorithm is probably the most intuitive of all statistical pattern recognition techniques.

Classification is based on a set of measurements of the objects (patterns) to classify that are deemed to carry discriminating information among them that define their membership to their respective classes. Such measurements are referred to as features and their full set for a given object as a feature vector.

Figure 3-14 shows the block diagram of a typical pattern recognition system. The sensor performs the measurements on the object under study. The feature extractor computes relevant features from those measurements and then the resulting feature vector is sent to the classifier for assigning the object under study to one among predefined classes.

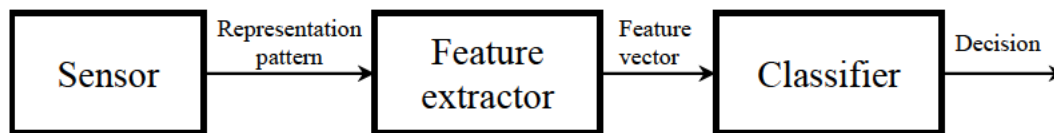


Figure 3-14: Typical pattern recognition system

3.5.2 Feature normalisation

Many classifiers base their decision on some distance measure between feature vectors. In order to prevent features with large numerical values from dominating in distance-based objective functions, normalisation of feature vectors of datasets is widely used [144].

A direct application of distance measures to features would implicitly assign bigger contributions to features with large ranges than to features with small ranges. Furthermore, features should be dimensionless, because dimensional features depend on units of measurements and therefore the choice of units of measurements may greatly affect the results of classification [144]. Hence, one should not use distance measures such as the Euclidean distance without normalisation of features [145].

Two kinds of normalisations have been discussed in the majority of papers, namely the min-max normalisation and the z-score standardisation [144].

The min-max normalisation normalises the data by dividing the feature component x_{ij} by its range using scaling with a shift as given by (3.118)

$$x_{ij}^* = \frac{x_{ij} - x_{\min,j}}{x_{\max,j} - x_{\min,j}} \quad (3.118)$$

where x_{ij}^* is the normalised feature value in the dataset, and $x_{\max,j}$ and $x_{\min,j}$ are respectively the maximum and minimum values of features F_j [144]. The results of scaling by (3.118) do not depend on the original data unit of measurements, and this linear scaling transforms the data to the range [0, 1].

The z-score standardisation transforms the feature component x_{ij} to a random variable with zero mean and unit variance as given by (3.119)

$$x_{ij}^* = \frac{x_{ij} - \mu_j}{\sigma_j} \quad (3.119)$$

where μ_j and σ_j are the sample mean and the sample standard deviation of the feature F_j respectively [145]. If we assume that each feature is normally distributed, the probability of x_{ij}^* being in the range [-1, 1] is 68%.

3.5.3 Euclidean distance classifier

For a Euclidean distance classifier, each pattern class ω_j is characterised by a vector \mathbf{m}_j which is the mean vector of the features of the training patterns of that class as described by (3.120)

$$\mathbf{m}_j = \frac{1}{N_j} \sum_{\mathbf{x} \in \omega_j} \mathbf{x} \quad (3.120)$$

where N_j is the number of training pattern vectors for class ω_j and the summation is taken over these vectors.

Determining the class membership of an unknown pattern with feature vector \mathbf{x} consists in computing the distance measures given by (3.121) and assigning \mathbf{x} to the class ω_i for which D_i is the smallest distance.

$$D_j(\mathbf{x}) = \|\mathbf{x} - \mathbf{m}_j\| \quad j=1, 2, \dots, W \quad (3.121)$$

where W is the number of classes.

Figure 3-15 illustrates the principle of the Euclidean in a two-dimensional feature space. In this case the test sample is classified as belonging to class 1 because the Euclidean distance from test point to mean class 1 is shorter than the Euclidean distance from test point to mean class 2.

The above procedure of training the Euclidean distance classifier is called maximum likelihood (ML) training [16]. In this procedure for each pattern class a characteristic feature vector Λ is the mean vector of the corresponding feature vectors in the training set.

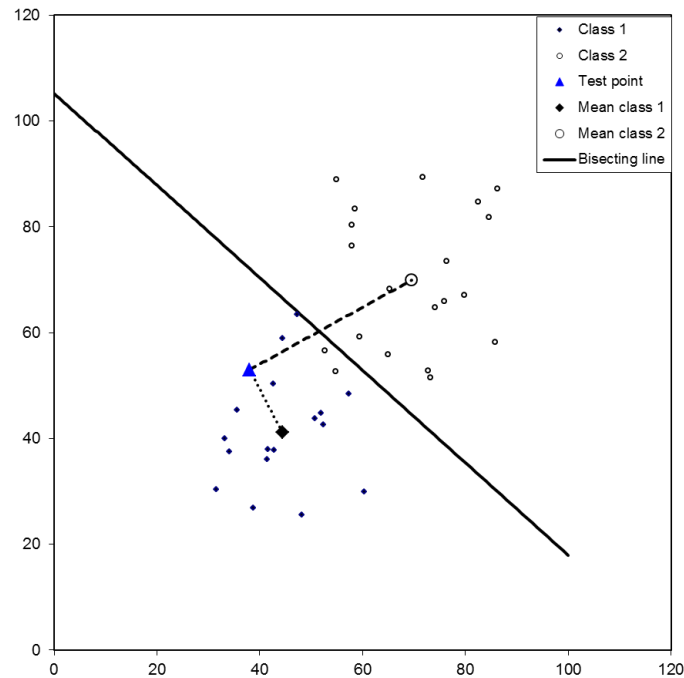


Figure 3-15: Principle of the Euclidean distance classifier

The minimum classification error (MCE) training [16, 146] allows one to find a better characteristic feature vector for each pattern class. The MCE training algorithm starts with the characteristic feature vectors obtained by the ML method and then adjusts them adaptively in order to achieve the highest classification rate of the feature vectors in the training set. That process is illustrated by Figure 3-16 for the defect detection experiments of this thesis.

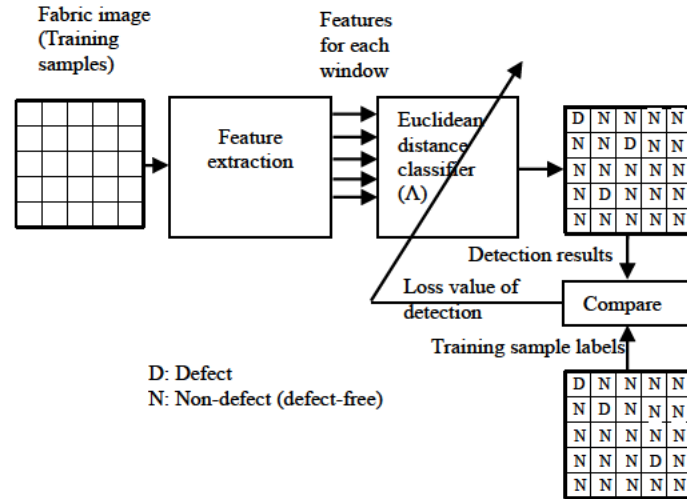


Figure 3-16: MCE training of an Euclidean distance classifier

The fabric images of the training set are submitted to a feature extraction operation. The features are extracted for each 32x32 window of the fabric image. The features of each window are compared to the characteristic feature vector (Λ) of the defective and defect-free classes and the window is classified as defective or defect-free. The detection results are then compared to the true training sample labels and the value of the detection loss function is evaluated. Using that loss value, the reference vectors (Λ) are adjusted in order to decrease the value of the loss function. New detection results are computed using the adjusted reference vectors and the new loss value of detection is evaluated. That process continues until the loss value of detection is minimised or is below a predefined threshold.

3.5.4 K-nearest-neighbour classifier

The K-nearest-neighbour classifier is a nonparametric classifier in that it does not depend on any assumption about the structure of the underlying density function [147].

The K-nearest-neighbour procedure for classifying a new feature vector sample X to one of the W classes ω_j is as follows [148]:

1. Determine the K-nearest feature vectors of the training pattern vectors using an appropriate distance metric.
2. Assign X to the class with the most representatives within the set of the K nearest neighbours.

There are several ways of breaking ties among competing classes;

- a) Ties may be broken arbitrarily.

- b) X may be assigned to the class of the nearest neighbour.
- c) X may be assigned to the class, out of the tying classes with k' votes each, that has the nearest mean vector to X (with the mean calculated over the k' samples)
- d) X may be assigned to the most compact class of the tying classes, i.e. the one for which the distance to the k^{th} member is the smallest.

The aspects that require being pre-specified for a K-nearest-neighbour classifier are (i) the number of neighbours K, (ii) the distance metric and (iii) the training dataset.

The number of neighbours K is selected as a trade-off between choosing a value large enough to reduce the sensibility to noise and choosing a value small enough that the neighbourhood does not extend to the domain of the other classes [148]. In this thesis, we will use $K=3$ for defect detection experiments (2-class problem) and $K=9$ for defect classification experiments (4-class problem).

The most commonly used metric in measuring the distance of a new sample from a training data feature vector is the Euclidean distance [148]. With such distance metric, feature normalisation is required as explained in Subsection 3.5.2.

3.5.5 Feed-forward neural network classifier

Figure 3-17 shows the architecture of a feed-forward neural network classifier with one hidden layer.

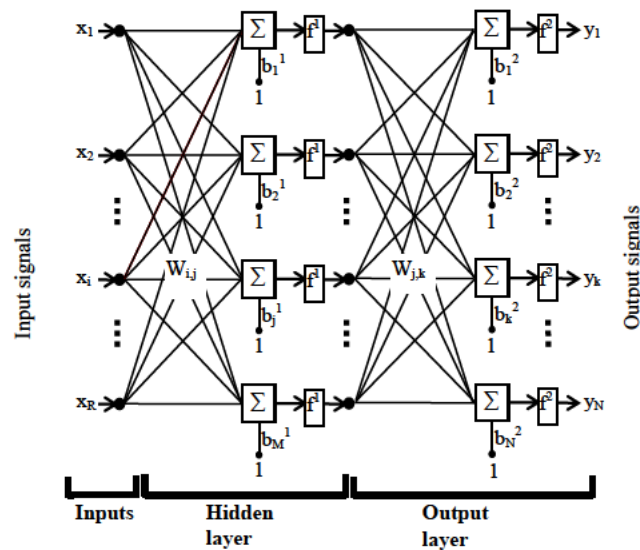


Figure 3-17: Feed-forward neural network with R inputs, M neurons in the hidden layer and N outputs [149]

The output of each neuron is computed taking the sum of the weighted inputs and the biases and passing it through the transfer function. The transfer function is typically a sigmoid function.

Among the algorithm used for supervised training, the backpropagation has emerged as the most widely used and successful for image processing [150]. During the training phase, the feature vectors of the training set are presented at the inputs of the network and the resulting outputs are compared to the desired outputs. Then using a gradient descent optimisation algorithm the weights and the biases of the network are adjusted to minimise the mean square error between the network outputs and the desired outputs.

Once trained, the network can be used to give reasonable answers when presented with inputs that the network has never seen.

The problem of choosing the neural network topology is mainly about deciding the number of hidden layers and the number of neurons in each of them. One hidden layer generally produces excellent results. In fact, for many practical problems, there is no reason to use more than one hidden layer [149, 151].

The number of neurons in the hidden layer must be chosen carefully. Using too few neurons in the hidden layers may result in underfitting. Underfitting occurs when there are too few neurons to adequately detect the signals in a complicated dataset. Using too many neurons in the hidden layer may result in overfitting. Overfitting occurs when the neuron network has so much training capacity that the limited amount of information in the training set is not enough to train all the neurons in the hidden layer. Furthermore, a large number of neurons in the hidden layers can increase the time it takes to train the network [151].

Heaton [151] provides some rule of thumb methods to determine the number of neurons to use in the hidden layer:

- The number of hidden neurons should be between the size of the input layer and the size of the output layer.
- The number of hidden neurons should be $\frac{2}{3}$ the size of the input layer, plus the size of the output layer.
- The number of hidden neurons should be less than twice the size of the input layer.

3.5.6 Multiclass classifiers

Several successful methods have been developed to solve the problem of binary classification where the objects to be classified belong to only two different classes. The multiclass classification is more delicate as many of the algorithms were basically introduced to solve binary classification problems [152].

Three main approaches are usually proposed for solving the multiclass classification problem: (i) natural extension of binary classifiers to the multiclass case, (ii) converting the multiclass classification problem into a set of binary classification problems that are efficiently solved using binary classifiers and (iii) the hierarchical division of the output space with classes arranged into a tree with a simple classifier at each node [152].

All three classifiers previously described in this thesis can be naturally extended and used for multiclass classification. Reference [152] provides more details about binary classifiers that can be naturally extended to the multiclass case.

Several methods have been proposed for the decomposition of multiclass classification problem into a set of binary classification problems. Among them, one can mention, the one-versus-all (OVA) [153], all-versus-all (AVA) [154], error-correcting output-coding (ECOC) [155] and generalised coding [156] approaches.

The OVA approach handles a K-class classification problem as follows. Each sample to classify is submitted to K binary classifiers with the K^{th} classifier trained with positive samples from class K and the negative samples from the other K-1 classes. When testing an unknown sample, the classifier producing the maximum output is considered the winner and its class label is assigned to the sample [152].

For the AVA approach, each class is compared to each other class. This requires $K(K-1)/2$ classifiers. When testing a new sample a voting is performed among the classifiers, and the class with the maximum votes wins [152].

The ECOC approach operates by training N binary classifiers to distinguish between K different classes. It uses a $K \times N$ binary matrix M. Each row of M corresponds to a certain class, while each column corresponds to a certain classifier. Value 1 indicates the class where training samples are used as positive training samples for the corresponding classifier, while value 0 indicates the class with negative training samples. When testing a new unseen sample, the output codeword from the N classifiers is compared to the given K codewords, and the one with the minimum hamming distance is considered the class label of the sample [152].

The generalised coding approach proposed by Allwein [156] is a unified approach that combines OVA, AVA and ECOC into a single strategy. Its coding matrix M is similar to the ECOC's, but it allows values in $\{-1, 0, +1\}$. The value of $+1$ in $M(k, n)$ indicates that samples belonging to class k are considered as positive samples to classifier n . A value of -1 denotes that those samples are considered as negative samples. A value of 0 instructs the classifier not to use the corresponding class. When testing an unknown sample, the codeword closest to the output of that sample is chosen as the class label.

3.6 Summary

In this chapter, we reviewed the main concepts about the methods that will be used in this thesis. Those concepts are divided into two groups, namely (i) concepts about feature extractors and (ii) concepts about classifiers.

The concepts about features extractors are (i) the grey level co-occurrence matrix, (ii) the wavelet transform and (iii) the Markov random fields, while the concepts reviewed about classifiers are those pertaining to classifiers we will use, with (i) the Euclidean distance classifier, (ii) the K-nearest-neighbour classifier and (iii) the feed-forward neural network classifier.

A grey level co-occurrence matrix allows characterising the texture of a given image by computing the joint probabilities of occurrence of pairs of pixels separated by a given distance in a given direction. The resulting information is represented in a matrix form. Once the grey level co-occurrence matrix is compiled, a certain number of statistics called GLCM features are computed from it and are used to characterise the texture. Haralick et al. [31] proposed fourteen such features, but later other researchers proposed some more. Because compiling the GLCM and extracting features from it is computationally expensive, ways to accelerate the process have been proposed. Among them we reviewed (i) reducing the number of grey levels of the original image, (ii) optimising the software implementation, (iii) using specialised hardware such as FPGA and (iv) using other representations of information, such as linked lists and tree data structures, instead of matrices.

Wavelet transforms allow decomposing a texture image at different resolutions. The resulting wavelet coefficients can be used to characterise the texture features in space and spatial frequency. In one dimension the wavelet transform of a signal is obtained by convolving the signal with a shifted and scaled version of the original (mother) wavelet. The result is the continuous wavelet transform and is highly redundant. The discrete wavelet transform is obtained by shifting and scaling the mother wavelet in discrete steps rather than continuously. The DWT can be efficiently computed by using the concept of filter banks.

The extension of the wavelet transform to two dimensions is done by performing the wavelet transform along the rows and then along the columns of the image. This results in an approximation sub-image A and three detail sub-images: vertical sub-image V_1 (with vertical edges but no horizontal edges), horizontal sub-image H_1 (with horizontal edges but no vertical edges) and diagonal sub-image V_1 (with no horizontal nor vertical edges).

The undecimated discrete wavelet transform is similar to DWT with the exception that the results of filtering the original signal are not downsampled. It has the advantage of being shift-invariant, but it is highly redundant. The dual-tree wavelet complex wavelet transform is less redundant and is almost shift-invariant. It is implemented using two real filter bank trees, one acting as the real part and the other as imaginary part of the complex wavelet transform. The filters used in the two filter bank trees should be designed so that the corresponding wavelets approximately form a Hilbert transform pair.

Once the wavelet transform of a texture image is computed, features are calculated from the resulting wavelet coefficients. The mean energy, the mean absolute value, the variance and the entropy of magnitudes of wavelet coefficients are the most frequently used.

Markov random fields (MRF) can be used to represent a fabric texture image by a stochastic model. The Markovian property is assumed. That means that the value of a pixel depends directly only on the values of neighbouring pixel. The size and shape of the assumed neighbourhood define the order of the model.

Markov random fields are equivalent to Gibbs random fields, which define the joint probability of all the pixels of an image in terms of energy function. Auto-logistic, auto-binomial, auto-normal and multilevel logistic are some of the types of MRF models that differ by how their corresponding energy function is defined.

Texture feature extraction using a MRF is equivalent to choosing a type of MRF model to use and then estimating its parameters for the specific texture image under study. The model parameters constitute the texture features. Coding, maximum pseudo-likelihood and least squares estimation are some of the methods available for parameter estimation of MRF models.

Once the features are extracted, using among others the above described methods, they are fed into a classifier for decision making. This may be, for example, declaring that a part of a fabric image is defective or not, or assigning a detected defect to one of the predefined classes.

Some types of classifiers are the Euclidean distance classifier, the K-nearest-neighbour classifier and feed-forward neural network classifier. All these classifiers need training before they can be used to classify unknown samples. Training means presenting to the classifiers features of known samples (known as training samples) along with their known true classes and letting the classifier memorise that information using a specific algorithm.

A Euclidean distance classifier is trained by calculating the class mean feature vector of the training samples for each class among those it can handle. This is called maximum likelihood (ML) training. Once thus trained, it can classify a new unknown sample by computing the Euclidean distance of its feature vector to each of the class mean feature vectors of the training samples. The sample is then assigned to the class that yields the shortest distance. An ML-trained Euclidean distance classifier can be made more effective by adaptively readjusting the class mean feature vectors so that the misclassification of training samples is minimised. This is called minimum classification error (MCE) training.

Training a K-nearest-neighbour classifier entails simply storing all the feature vectors of the training samples along with their true class labels. To classify an unknown new sample, the distance between its feature vector and each of the feature vectors of the training samples is calculated. The K training samples with the shortest distances are identified. The sample is then assigned to the class that has the highest number of representatives among the K nearest neighbours of the unknown sample.

A feed-forward neural network classifier is made up of artificial neural cells modelled after neurons of the human brain. The neural cells are arranged in interconnected layers of neural cells that process information in parallel. The feed-forward neural network classifier is usually made up of an input layer, one or more hidden layers and an output layer. The number of inputs is equal to the number of features in the feature vector and each input receives a feature value. The values of input features are individually weighted and fed into the neural cells of the hidden layer. Finally, the weighted outputs of neural cells of the hidden layer are fed into the neural cells of the output layer. The cells in the output layers are properly coded to represent the classes involved in the classification task.

When training a feed-forward neural network, the feature vectors of the training samples are consecutively presented at the inputs, then for each feature vector the resulting output of the neural network is compared to the desired output (corresponding to the class label of the input vector). The weights of the network are then adjusted according to an algorithm known as backpropagation in order to usually minimise the mean square error between the obtained

outputs and the desired outputs for the samples of the training set. Once trained, the feed-forward neural network is able to classify unknown new samples.

Multiclass classification can be done either directly or by combining many binary classifiers.

Chapter 4: Description of the experimental data

4.1 Introduction

The experiments pertaining to this thesis were done using the images of the standard dataset TILDA. It is a textile texture database developed by the Texture Analysis working group of the Deutsche Forschungsgemeinschaft, major research programme “Automatic Visual Inspection of Technical Objects” from the Technical University of Hamburg, Germany.

One of the main advantages of using that dataset is the availability of a large number of samples from different fabric categories. That ensures high validity and reliability of the results of the tests done on it and allows expanding the scope of applicability of the derived conclusions. Furthermore, significant research work has been done using the entire or a part of that dataset [35, 53, 83, 134, 157-172]. That makes it possible to compare results from different researchers and from different methods.

The dataset contains pictures from four different classes of fabrics defined by the regularity of the surface structure. Two representatives of each type are included. Figure 4-2 shows sample pictures of the two representatives for each of the four fabric classes. Fifty images per representative of defect-free surface are present in the dataset. Furthermore, seven defect types are defined within each class of fabric and the dataset contains fifty images representative of each type of defect. We will deal with only four of the seven types of defects because we realised that only those four types correspond to real fabric defects, while the three others correspond to image deformation due to lighting and camera tilting. Figure 4-1 shows the structure of the dataset we used in our experiments, while Table 4-1 shows the detailed numbers of images from the dataset that we considered in this thesis.

All images are monochrome with 256 grey levels. They are sized 768x512 pixels and are stored in the TIFF format. The images are arbitrarily rotated. This makes the methods of defect detection more difficult, but the successful detection methods become more robust.

To facilitate the comparison of the dataset to the defect classes used in the textile industry standards, an additional twenty images from each category of commercially available fabric defects are presented. The dataset contains a total of 3 228 images.

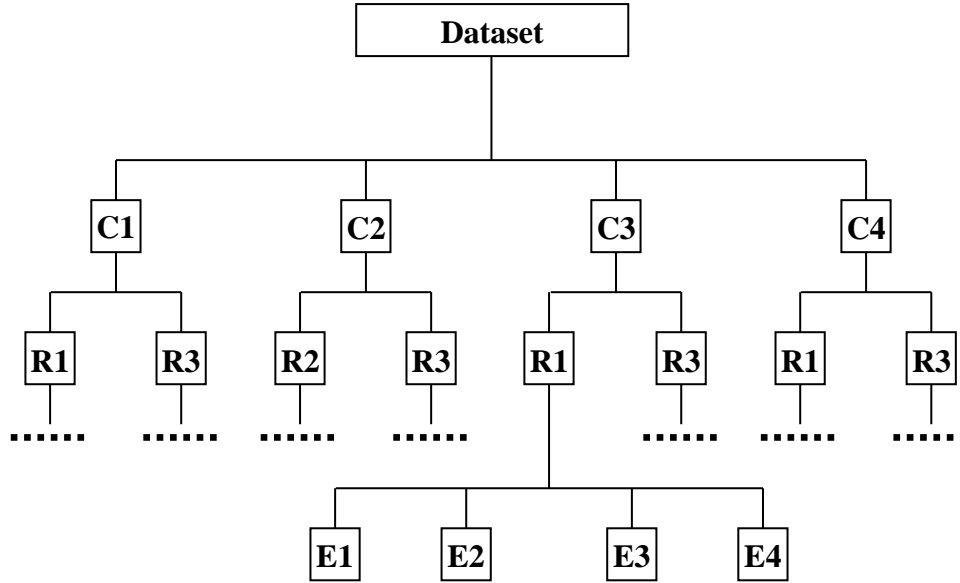


Figure 4-1: Structure of the dataset used in this thesis

Table 4-1: The TILDA images used in this thesis

Fabric class	Representative	Error type	Number of images	Fabric class	Representative	Error type	Number of images
C1	R1	E1	50	C3	R1	E1	50
		E2	50			E2	50
		E3	50			E3	50
		E4	50			E4	50
	R3	E1	50		R3	E1	50
		E2	50			E2	50
		E3	50			E3	50
		E4	50			E4	50
C2	R2	E1	50	C4	R1	E1	50
		E2	50			E2	50
		E3	50			E3	50
		E4	50			E4	50
	R3	E1	50		R3	E1	50
		E2	50			E2	50
		E3	50			E3	50
		E4	50			E4	50

4.2 The fabric classes and defect types

The four classes of fabrics present in the dataset are shown in Table 4-2.

Table 4-2: Fabric classes in the TILDA dataset

Class	Description
Class 1 (C1)	Very fine fabrics with or without visible external structure. Examples are non-printed rayon or silk. The first row of Figure 4-2 shows some examples.
Class 2 (C2)	Fabrics with a low variance stochastic structure. The surface contains no imprints. Examples are wool or jute. The second row of Figure 4-2 shows some representatives.
Class 3 (C3)	Fabrics with a periodic structure clearly visible. Examples are fabrics with a printed diamond pattern or curtains. Some representatives of that class are shown in the third row of Figure 4-2.
Class 4 (C4)	Printed fabrics with no apparent periodicity. An example is viscose printed with some flowers of varying girth. The last row of Figure 4-2 shows some examples.

The defect types used in the dataset were inspired by the approach proposed by the Institute of Textile Machinery and Textile Industry of the ETH Zurich [173]. A brief description of the observed defects included in the dataset is shown in Table 4-3.

Table 4-3: Fabric defects observed in the TILDA dataset

Defect type	Description
Defect E0	No defect in the fabric.
Defect E1	Holes in the fabric and cuts caused by mechanical damage.
Defect E2	Oil stain and colour fading.
Defect E3	Thread error. Condensations of filaments (not mechanically induced cracks). Absence of individual threads in the fabric.
Defect E4	Foreign body in the fabric (so-called flight)
Defect E5	Wrinkles in the fabric (with no mechanical damage).
Defect E6	Changing lighting conditions.
Defect E7	Affine distortion by tilting the camera and changing the distance between the camera and the specimen.

As explained earlier, only defect types E0 through E4 will be considered in this thesis.

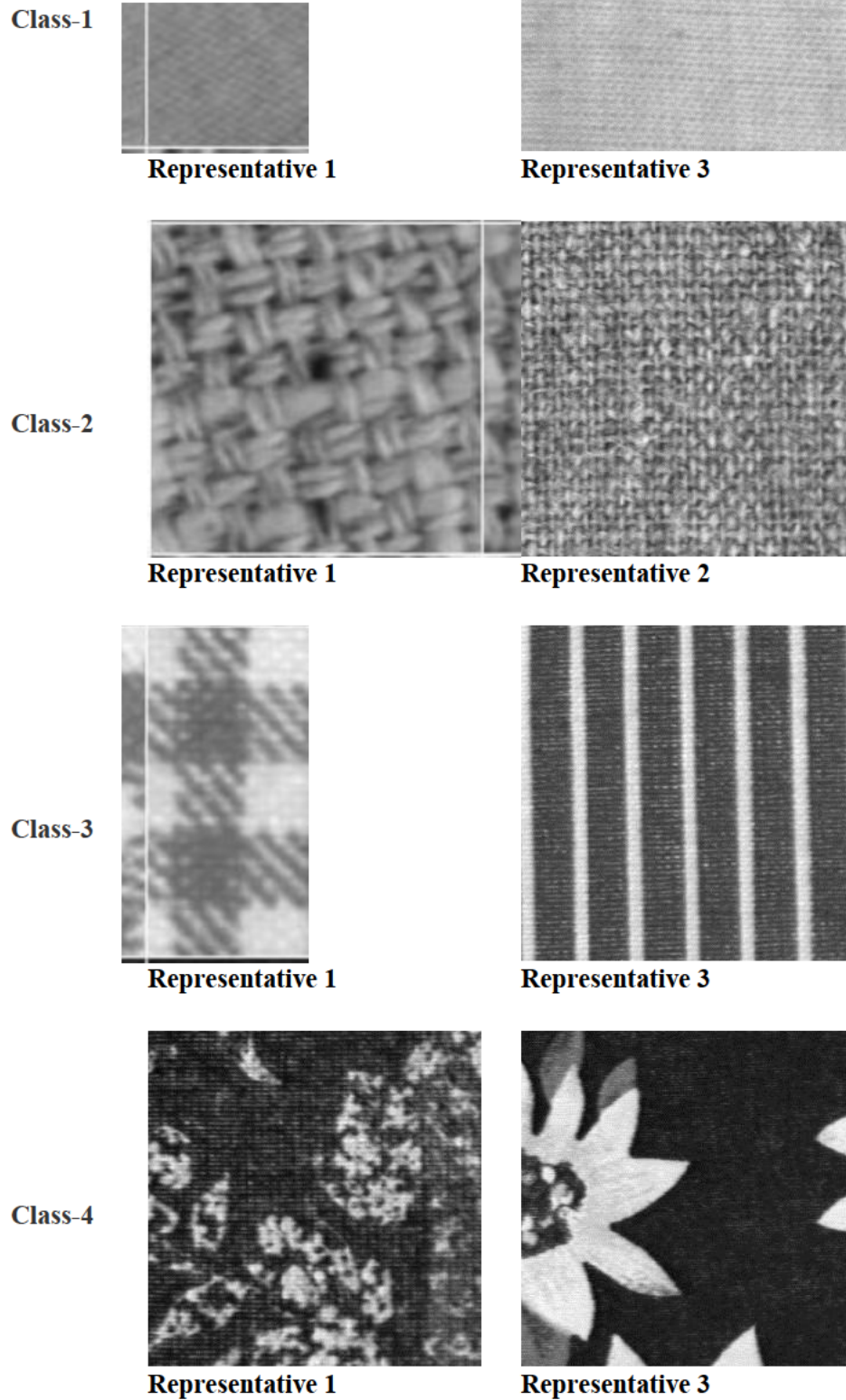


Figure 4-2: Representatives of the fabric classes contained in the TILDA dataset

4.3 Elementary experimental sample

For all the defect detection or classification experiments of this thesis, the fabric image is divided into non-overlapping windows and then each window is independently classified as defective or defect-free. If the window is detected as defective it may be assigned to one of the four defect categories defined in Section 4.2.

The choice of window size depends mainly on two considerations:

- i) The accuracy in localising the defects. The smaller the window, the more localised the defect.
- ii) The degree of representation of the texture by the data inside the window. The bigger the window, the more the data it contains represents the texture.

A trade-off between the two considerations should be found. Yang [16] performed experiments on fabric images for the purpose of defect detection using window sizes of 16x16, 32x32 and 64x64 pixels, and found 32x32 to be the best window size. Several other researchers [20, 8] used a 32x32 window which seems to be a good compromise between the two above-mentioned conflicting considerations. Therefore we will use 32x32 windows. Each fabric image will be divided into 32x32 non-overlapping windows and the elementary experimental sample will be each of those windows. Figure 4-3 illustrates the division of each image.

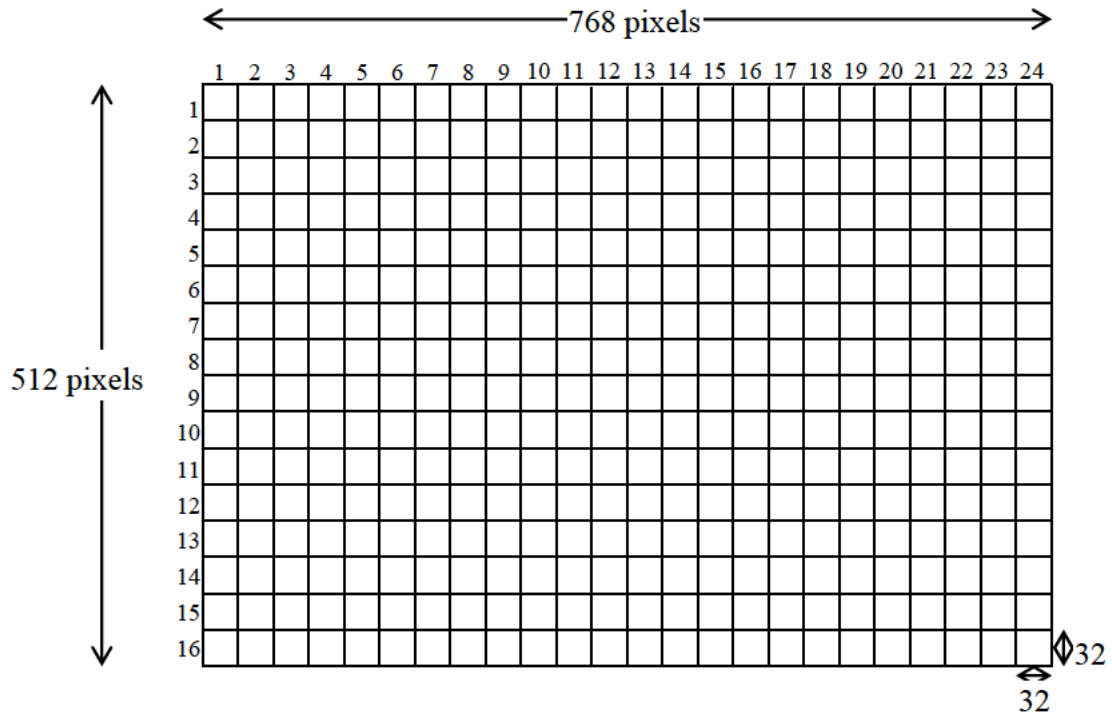


Figure 4-3: Division of a fabric image into elementary samples

4.4 Labelling of the samples

The ground truth information in TILDA is limited to an indication that a fabric image is either defect-free or defective. That information is encoded in the name of the folder in which the image file is stored, E0 for defect-free images, E1 for images with defects of type E1, and so on. No information about the number, the location or the size of defects in any given image is available. Yet, that information is required for classifier training and for the performance assessment of either the classifier or the overall method of detection or classification of fabric defects. That is the reason why we had to devise a semi-automatic way of visually assessing and labelling each of the elementary samples of the dataset (each window of 32x32 pixels of every image as shown in Figure 4-3) as either defective or defect-free.

If the sample is defect-free it is assigned label 0, otherwise it is assigned label 1, 2, 3 or 4 depending on the type of defect it contains as described in Section 4.2. Therefore a defective sample from an image stored in a folder named E1 is assigned label 1 and similarly defective samples from images stored in folders named E2, E3 or E4 are assigned labels 2, 3 or 4 respectively.

Given the huge amount of image data, the algorithm should be able to simultaneously process many neighbouring samples with the same label. The proposed algorithm in pseudo-code is

shown in Table 4-4. The reader should refer to Figure 4-1 for the folder names used in the algorithm.

The algorithm was implemented using MATLAB and steps 5 and 7 were implemented using the MATLAB function ‘*roipoly*’. The choice of the defective area label in steps 7 and 9 is done according to the folder name where the image under consideration is stored (label 1 for images in folder E1 and labels 2, 3 or 4 for images in folders E2, E3 and E4 respectively). The threshold number (40) of defective pixels to decide whether a 32x32 window is defective or not was chosen empirically by looking at the subdivided image and the labelling results.

Table 4-4: Algorithm for sample labelling

1)	For each of the eight fabric types (Folders C1\R1, C1\R3, C2\R2, C2\R3, C3\R1, C3\R3, C4\R1 and C4\R3)
2)	For each of the four defect types (Folders E1, E2, E3 and E4)
3)	For each of the 50 images
4)	Display the image to the user.
5)	Allow the user to encircle the defective area using the mouse.
6)	If there is another defective area go to step 5.
7)	Create a binary image of the same size as the original image where the pixels corresponding to defective areas have the value of the defect type number (1, 2, 3 or 4) and the remaining pixels have the value 0.
8)	Divide the binary image obtained in step 7 into 32x32 windows (we obtain a 16x24 array of such windows).
9)	Create a 16x24 label matrix and set each label to defective (1, 2, 3 or 4) if the number of defective pixels is greater than 40 in the corresponding window. Otherwise set the label to defect-free (0).
10)	Save both the binary image obtained in step 7 and the label matrix obtained in step 9.
11)	Continue with the next image
12)	Continue with the next defect type
13)	Continue with the next fabric type

4.5 Training and testing data sets

The experimental samples we used in our experiments is a subset of all the elementary experimental samples as defined in Section 4.3 obtained from all the 1,600 images. The samples were chosen as follows. All the defective samples were included because the defective samples are much fewer than defect-free samples. The number of defective samples was 37,546. We decided to include the same number of defect-free samples chosen as

follows. From each image, the same number of defect-free as defective samples was taken. The defect-free samples were chosen randomly, but once selected they were recorded so that the same samples were used in all the experiments. That was done in order to make possible the comparison of the experimental results.

After setting up the experimental dataset as described in the previous paragraph, it was divided into two parts of almost the same size, the training data set for training the classifiers and the testing dataset for assessing the performance of the trained classifier and the overall detection or classification system. One half of the defective samples from every image was included in the training set while the other half was included in the testing set. When the number of defective samples from an image was odd, the additional sample was included in the testing set. The same process was applied to the defect-free samples. Table 4-5 shows the detailed numbers of samples in the experimental dataset. In the table, C1 through C4 refer to the four classes of fabrics, E1 through E4 to the four types of defects as described in Table 4-2 and Table 4-3, while TRN and TST refer to the training and testing set respectively.

Table 4-5: Number of samples in the experimental dataset

		C1		C2		C3		C4		Total
		C1R1	C1R3	C2R2	C2R3	C3R1	C3R3	C4R1	C4R3	
E1	TRN	480	576	830	1812	1000	526	800	432	6456
	TST	532	636	880	1870	1060	586	854	480	6898
	Total	1012	1212	1710	3682	2060	1112	1654	912	13354
E2	TRN	1170	700	1268	1748	1398	1130	1626	692	9732
	TST	1208	744	1326	1780	1448	1180	1680	722	10088
	Total	2378	1444	2594	3528	2846	2310	3306	1414	19820
E3	TRN	806	1236	1568	2492	1190	670	868	598	9428
	TST	836	1282	1632	2532	1236	722	926	648	9814
	Total	1642	2518	3200	5024	2426	1392	1794	1246	19242
E4	TRN	1468	1474	808	1912	1724	1310	1344	1106	11146
	TST	1496	1532	858	1960	1782	1362	1390	1150	11530
	Total	2964	3006	1666	3872	3506	2672	2734	2256	22676
Total	TRN	3924	3986	4474	7964	5312	3636	4638	2828	36762
	TST	4072	4194	4696	8142	5526	3850	4850	3000	38330
	Total	7996	8180	9170	16106	10838	7486	9488	5828	75092

4.6 Performance evaluation

In this section we define the performance measures for defect detection and defect classification used in this thesis. Let us denote by TP the number of defective samples

detected as defective, by TN the number of defect-free samples correctly detected as defect-free, by FP the number of defect-free samples incorrectly detected as defective and by FN the number of defective samples incorrectly detected as defect-free.

The detection rate is the ratio of the number of samples correctly detected to the total number of samples as expressed in (4.1). That ratio is normally expressed as a percentage.

$$\text{Detection rate} = \frac{\text{Number of samples correctly detected}}{\text{Total number of samples}} \quad (4.1)$$

The false alarm rate is the ratio of the number of non-defective samples detected as defective to the total number of the non-defective samples. This can be expressed by (4.2). In the similar way, that ratio is normally expressed as a percentage.

$$\text{False alarm rate} = \frac{\text{FP}}{\text{TN} + \text{FP}} \quad (4.2)$$

In defect classification experiments, we used the classification rate defined as the ratio of the number of samples correctly classified into their respective classes to the total number of samples. It is to be noted that the defect classification rate depends on the number of defect classes involved in the experiment. The more defect classes there are, the more likely the samples will be misclassified and the lower will be the classification rate for a given defect classification method.

The confusion matrix allows representing the classification performance in a summarised form as shown in Table 4-6.

Table 4-6: Confusion matrix

		True class			
		ω_1	ω_2	...	ω_N
Predicted class	ω_1	e_{11}	e_{12}	...	e_{1N}
	ω_2	e_{21}	e_{22}	...	e_{2N}

	ω_N	e_{N1}	e_{N2}	...	e_{NN}

The correct classification rate can be expressed as (4.3)

$$\text{Classification rate} = \frac{\sum_{i=1}^N e_{ii}}{\sum_{i=1}^N \sum_{j=1}^N e_{ij}} \quad (4.3)$$

where N is the number of classes and e_{ij} is the number of samples.

4.7 Summary

The experiments of this thesis were performed using textile images from the TILDA dataset, a textile texture database developed by the Textile Analysis working group of the Deutsche Forschungsgemeinschaft in Germany.

The dataset contains pictures of fabrics grouped into four different classes with two representatives per class. This resulted into eight different types of fabrics in the dataset. The fabric images contain defects of seven different types, but only four of them were used in this thesis.

For the experiments for this thesis, the fabric images were divided into non-overlapping windows of size 32x32 pixels each. Those windows individually constitute our experimental samples.

Since no labelling information (as defective or defect-free) of those experimental samples is directly available in the TILDA dataset, a semi-automatic way of visually assessing and labelling them was devised and implemented using MATLAB.

The resulting final experimental dataset was made of 75,092 experimental samples (32x32 image patches) properly labelled, extracted from eight different types of fabrics and containing defects of four different classes in addition to the defectless class. Those samples were then divided equally into the training and testing datasets.

For fabric defect detection and classification performance evaluation, we defined and used the following metrics: (i) the detection rate, (ii) the false alarm rate, (iii) the classification rate and (iv) the confusion matrix.

Chapter 5: Exploration of grey level co-occurrence matrix features

5.1 Introduction

The objective of this chapter is to identify the most effective co-occurrence matrix features and to devise a fast algorithm to implement them. Various research results indicate that co-occurrence matrix features are effective in discriminating texture [10, 30, 43]. However, the same research results point out that the extraction of those features is computationally expensive. Therefore, we want to identify the lowest number of those features that preserve the texture discriminating ability of the full set of the features. The assumption is that the fewer the features the less expensive they are to compute.

The co-occurrence feature extraction from a texture image normally requires two phases, the compilation of the co-occurrence matrix and the computation of the features from the co-occurrence matrix. As presented in subsection 3.2.4, Clausi and Zhao [120] devised the GLCHS representation that allows faster computation of the co-occurrence features. We propose a modified-version of GLCHS for implementation in MATLAB and use it for all our co-occurrence feature experiments.

This chapter is organised as follows. In Section 5.2 we determine the number of optimal GLCM features when an ML-trained Euclidean distance classifier is used. In Section 5.3 we identify the actual GLCM features for the different types of fabrics when an ML-trained Euclidean distance classifier is used. Section 5.4 studies the variation of defect detection performance of the GLCM features as the quantisation level of the fabric images changes, while Section 5.5 studies that variation as the intersample distance parameter varies. In Section 5.6 we investigate the defect detection performance of GLCM features when the MCE-trained Euclidean distance, the K-nearest-neighbour (K=3) and the feed-forward neural network classifiers are used. Section 5.7 deals with the problem of defect classification using GLCM features, while Section 5.8 deals with the modified GLCHS algorithm for accelerating the computation of the GLCM features. We summarise the chapter in Section 5.9.

5.2 Optimal number of the GLCM features for fabric defect detection

5.2.1 Introduction

Haralick et al. [31] proposed fourteen GLCM features for texture discrimination as described in Subsection 3.2.3. Normally those features are computed from co-occurrence matrices compiled for four different directions. The images from which our experimental samples were extracted were arbitrarily rotated and consequently the relative orientation of samples with

respect to each other is unknown. The advantage of using such experimental samples is that the defect detection methods that perform well on them are more robust. However, that requires the textural features used to be rotation invariant.

We adopted the method proposed by Haralick et al. [31] to get almost rotation invariant GLCM features. Firstly, we compiled four symmetric co-occurrence matrices P_0 , P_{45} , P_{90} and P_{135} corresponding respectively to 0° , 45° , 90° and 135° for each of the experimental samples. Secondly, we extracted the fourteen features $f_i^1, f_i^2, \dots, f_i^{14}$ ($i=0, 45, 90, 135$) from each of the four co-occurrence matrices. Finally, we calculated the average $f^j = \frac{f_0^j + f_{45}^j + f_{90}^j + f_{135}^j}{4}$ and the range $f^{j+14} = \max(f_0^j, f_{45}^j, f_{90}^j, f_{135}^j) - \min(f_0^j, f_{45}^j, f_{90}^j, f_{135}^j)$ ($j=1, 2, \dots, 14$) of the same feature from the four co-occurrence features for each of the fourteen features. That led for each experimental sample to 28 features – fourteen average features and fourteen range features – which are invariant to any rotation by 0° , 45° , 90° , 135° , 180° , 225° , 270° or 315° . The four last angle measures are justified by the symmetry of the co-occurrence matrix.

It has also been shown that many of those features are correlated among each other [48]. Therefore, the objective of the experiment under this section is to find for each fabric type, the number of features that yields the best detection rate. As we have four classes of fabrics and two particular representatives of each class, the experiment was performed for each of the eight resulting fabric types. Those types were referred to using the names of the folders in which the corresponding images are stored as C1R1, C1R3, C2R2, C2R3, C3R1, C3R3, C4R1 and C4R3. That can be made clear by referring to Figure 4-1.

The average feature and the range feature of the same feature were always considered together and we refer to such a pair as combined feature in this experiment. Therefore each sample has a feature vector made up of fourteen combined features. The objective of the current experiment is to find the optimal number of combined features for defect detection and identify for each fabric type the optimal combined features.

Firstly, we evaluate the defect detection performance as the number of combined features considered increases. We find the best detection rate that can be achieved using a combined-feature for each of the eight fabric types and for each of the four defect types. We then do the same for pairs of combined features and then for triplets and so forth until reaching all the fourteen combined features. As there are fourteen combined features, the number of combinations to be tried for each fabric type and each error type is only $\sum_{k=1}^{14} \binom{14}{k} = \frac{14!}{k!(14-k)!} = 16383$ and an exhaustive search was possible and therefore was performed.

5.2.2 Description of the experiment

The steps of the experiment for each of the eight fabric types were as follows.

1. For each of the samples in both the training and testing sets
 - a. Compile the four co-occurrence matrices
 - b. Calculate the fourteen features from each of the four co-occurrence matrices using the formulae provided in Subsection 3.2.3.
 - c. Calculate the averages and the ranges of the fourteen features over the four co-occurrence matrices. That leads to a feature vector made up of fourteen average features numbered from 1 to 14 and fourteen range features numbered from 15 to 28.
2. Once all the features are calculated, arrange the feature vectors per defect type into E1, E2, E3 and E4 groups. That is done for both the training and testing sets. That leads to an overall number of 64 groups as shown in Table 5-1.

Table 5-1: Groups of samples and their respective number of elements

Training set			Testing set		
S/N	Group	Number of samples	S/N	Group	Number of samples
1	C1R1E1	480	33	C1R1E1	532
2	C1R1E2	1170	34	C1R1E2	1208
3	C1R1E3	806	35	C1R1E3	836
4	C1R1E4	1468	36	C1R1E4	1496
5	C1R3E1	576	37	C1R3E1	636
6	C1R3E2	700	38	C1R3E2	744
7	C1R3E3	1236	39	C1R3E3	1282
8	C1R3E4	1474	40	C1R3E4	1532
9	C2R2E1	830	41	C2R2E1	880
10	C2R2E2	1268	42	C2R2E2	1326
11	C2R2E3	1568	43	C2R2E3	1632
12	C2R2E4	808	44	C2R2E4	858
13	C2R3E1	1812	45	C2R3E1	1870
14	C2R3E2	1748	46	C2R3E2	1780
15	C2R3E3	2492	47	C2R3E3	2532
16	C2R3E4	1912	48	C2R3E4	1960
17	C3R1E1	1000	49	C3R1E1	1060
18	C3R1E2	1398	50	C3R1E2	1448
19	C3R1E3	1190	51	C3R1E3	1236
20	C3R1E4	1724	52	C3R1E4	1782
21	C3R3E1	526	53	C3R3E1	586
22	C3R3E2	1130	54	C3R3E2	1180
23	C3R3E3	670	55	C3R3E3	722

Training set			Testing set		
S/N	Group	Number of samples	S/N	Group	Number of samples
24	C3R3E4	1310	56	C3R3E4	1362
25	C4R1E1	800	57	C4R1E1	854
26	C4R1E2	1626	58	C4R1E2	1680
27	C4R1E3	868	59	C4R1E3	926
28	C4R1E4	1344	60	C4R1E4	1390
29	C4R3E1	432	61	C4R3E1	480
30	C4R3E2	692	62	C4R3E2	722
31	C4R3E3	598	63	C4R3E3	648
32	C4R3E4	1106	64	C4R3E4	1150

- Organise the feature vectors of each group in a feature table. Table 5-2 shows the example of such a table for the group C1R1E1 of the training set. Each row contains all the features (feature vector) of a particular sample while each column contains the values of a particular feature for all the samples of that group. Besides the feature vector, a label is associated to each sample, 0 if the sample is defect-free and 1, 2, 3 or 4 if the sample is defective and belongs to the group E1, E2, E3 or E4 respectively.

Table 5-2: Feature table of the samples of C1R1E1 group of the training set

	Feature 1	Feature 2	...	Feature j	...	Feature 28
Sample 1	$f_{1,1}$	$f_{1,2}$...	$f_{1,j}$...	$f_{1,28}$
Sample 2	$f_{2,1}$	$f_{2,2}$...	$f_{2,j}$...	$f_{2,28}$
...
Sample i	$f_{i,1}$	$f_{i,2}$...	$f_{i,j}$...	$f_{i,28}$
...
Sample 480	$f_{480,1}$	$f_{480,2}$...	$f_{480,j}$...	$f_{480,28}$

- Normalise the features of each of the groups of the training set (groups numbered 1 to 32) using the min-max standardisation [145] as described in Subsection 3.5.2 according to the formula given by (5.1)

$$f_{i,j}^* = \frac{f_{i,j} - f_{\min j}}{f_{\max j} - f_{\min j}} \quad (5.1)$$

Where $f_{i,j}$ is the original feature value, $f_{i,j}^*$ the normalised feature value, and $f_{\min j}$ and $f_{\max j}$ are the minimum and the maximum value of feature j in the training set of the group.

- Normalise the features of each of the groups of testing set (groups numbered 33 to 64) using the min-max standardisation. Use (5.1) as well, but in this case $f_{\min j}$

and $f_{\max j}$ refer to the minimum and the maximum value of the feature j in the corresponding training set group rather than the current testing set.

6. For each of the groups in the training set, train a Euclidean distance classifier using the normalised feature vectors. As described in Subsection 3.5.3, that means calculating two mean normalised feature vectors, one of feature vectors of samples labelled as defect-free (label 0) and one of normalised feature vectors of samples labelled as defective (label 1, 2, 3 or 4). That can be expressed by (5.2)

$$\mathbf{m}_k = \frac{1}{N_k} \sum_{\mathbf{F} \in \omega_k} \mathbf{F}, \text{ with } \mathbf{F} = [f_{i,1}^*, f_{i,2}^*, f_{i,3}^*, \dots, f_{i,28}^*] \quad (5.2)$$

where $k \in \{0, l\}$ with $l=1, 2, 3, 4$ and N_k is the number of training sample of the class ω_k .

7. For each of the 64 groups in both the training and testing set, perform the Euclidean distance classification of the normalised features vectors considering in each case one combined feature and record the highest detection rate achieved as well as the combined feature used to achieve it. This procedure is detailed in pseudo-code in Table 5-3.

Table 5-3: Algorithm for evaluating the highest detection performance obtained with a single GLCM feature

1)	For each of the eight fabric types (C1R1, C1R3, C2R2, C2R3, C3R1, C3R3, C4R1 and C4R3)
2)	For each of the four defect types (E1, E2, E3 and E4)
3)	For each of the two sample sets (training and testing)
4)	Perform the Euclidean distance based detection using only one of the fourteen combined features.
5)	Record the highest detection rate obtained in step 4 as well as the corresponding combined feature.
6)	Continue with the next sample set
7)	Continue with the next defect type
8)	Continue with the next fabric type

8. For each of the 64 groups in both the training and testing set, perform the Euclidean distance classification of the normalised features vectors considering in each case a pair of combined features and record the highest detection rate

achieved, as well as the pair of combined features used to achieve it. The detailed procedure is similar to the one illustrated in Table 5-3, with the difference that in step 4) we consider each time one of the 91 possible pairs of combined features. Note that as there are fourteen combined features, the number of possible pair combinations is $\binom{14}{2} = 91$.

9. Repeat step 8, but use triplets of combined features instead of pairs. There are 364 possible triplets of combined features.
10. Do the same for quadruplets of features, then quintuplets and so forth until all fourteen combined features are used.

5.2.3 Results and interpretation

Table 5-4 shows the best detection rate obtained for the n-tuplets of combined features for all the 64 groups of samples (or feature vectors), as well as the mean of the best detection rates for all the groups. Here n refers to the number of combined features. Figure 5-1 graphically illustrates how the mean of the best detection rates changes as the number of combined features increases.

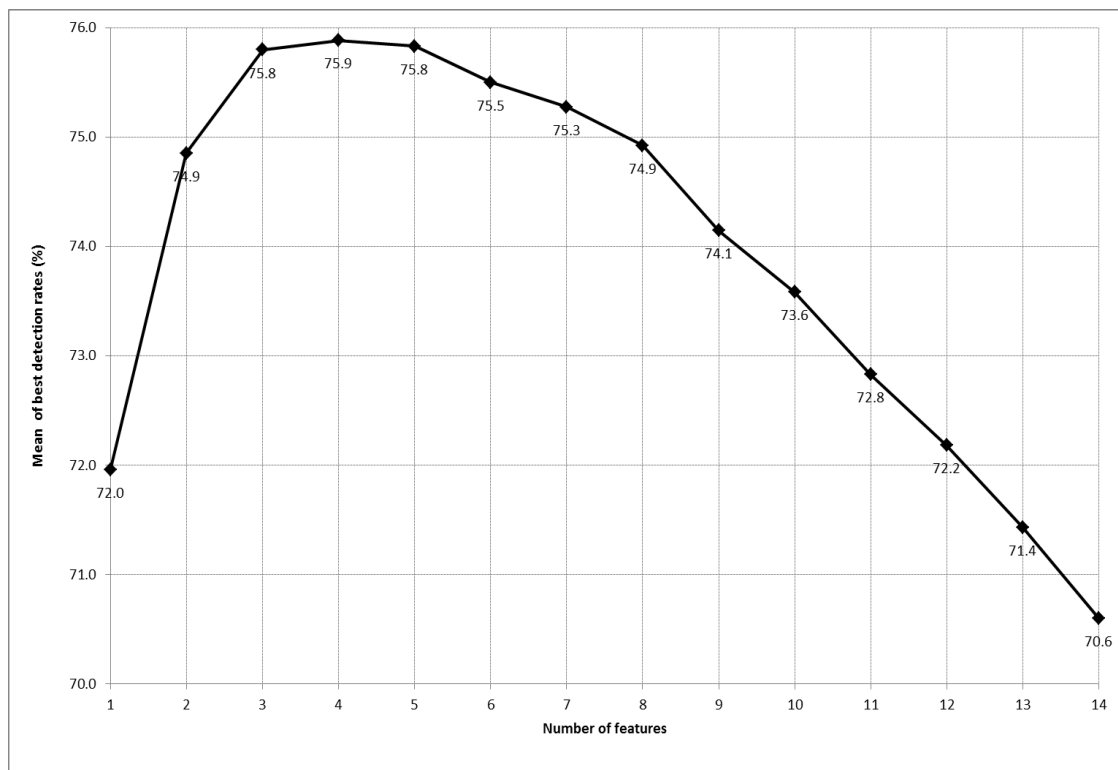


Figure 5-1: Mean of best detection rates as function of the number of GLCM features for all the samples in the dataset

There is a significant improvement of the detection rate (about 3%) as the number of considered combined features is increased from one to two. There is additional improvement but lesser in magnitude as the number of combined features is increased from two to three. There is practically no improvement as the number of combined features is changed from three to four or from four to five. The mean value of best detection rates drops as the number of combined features is increased beyond five.

Looking at Figure 5-2 that shows in graph how the averages of the best detection rate changes as the number n of combined features increases, one can see that the trend of variation of the best detection rate with respect to the number of combined features, is generally kept for the individual fabric types.

Table 5-4: Average of best detection rates as function of the number of GLCM features

	Fabric types								MEAN VALUE
Number of features	C1		C2		C3		C4		
	R1	R3	R2	R3	R1	R3	R1	R3	
1	88.6	84.8	75.3	71.5	65.1	68.3	60.6	61.3	72.0
2	90.3	86.1	79.2	74.3	70.6	70.6	65.2	62.6	74.9
3	90.6	86.4	81.2	75.1	73.1	71.6	65.6	62.8	75.8
4	90.7	86.5	81.6	75.5	73.7	70.7	66.1	62.4	75.9
5	90.7	86.5	81.6	75.8	73.7	70.3	65.8	62.2	75.8
6	90.6	86.4	81.1	75.9	73.4	69.2	65.3	62.0	75.5
7	90.6	86.5	80.9	75.9	72.8	68.8	65.4	61.2	75.3
8	90.5	86.4	80.6	75.8	71.8	68.1	65.2	60.9	74.9
9	90.3	86.3	80.2	75.7	70.7	67.4	62.1	60.5	74.1
10	89.9	86.2	79.9	75.4	69.3	66.6	61.1	60.2	73.6
11	89.6	86.0	79.3	74.8	67.7	64.9	60.6	59.8	72.8
12	89.3	85.7	78.7	74.3	66.4	63.6	60.2	59.3	72.2
13	88.7	85.4	78.1	73.5	65.0	62.5	59.6	58.6	71.4
14	87.9	84.7	77.2	72.7	63.7	61.4	59.2	58.1	70.6

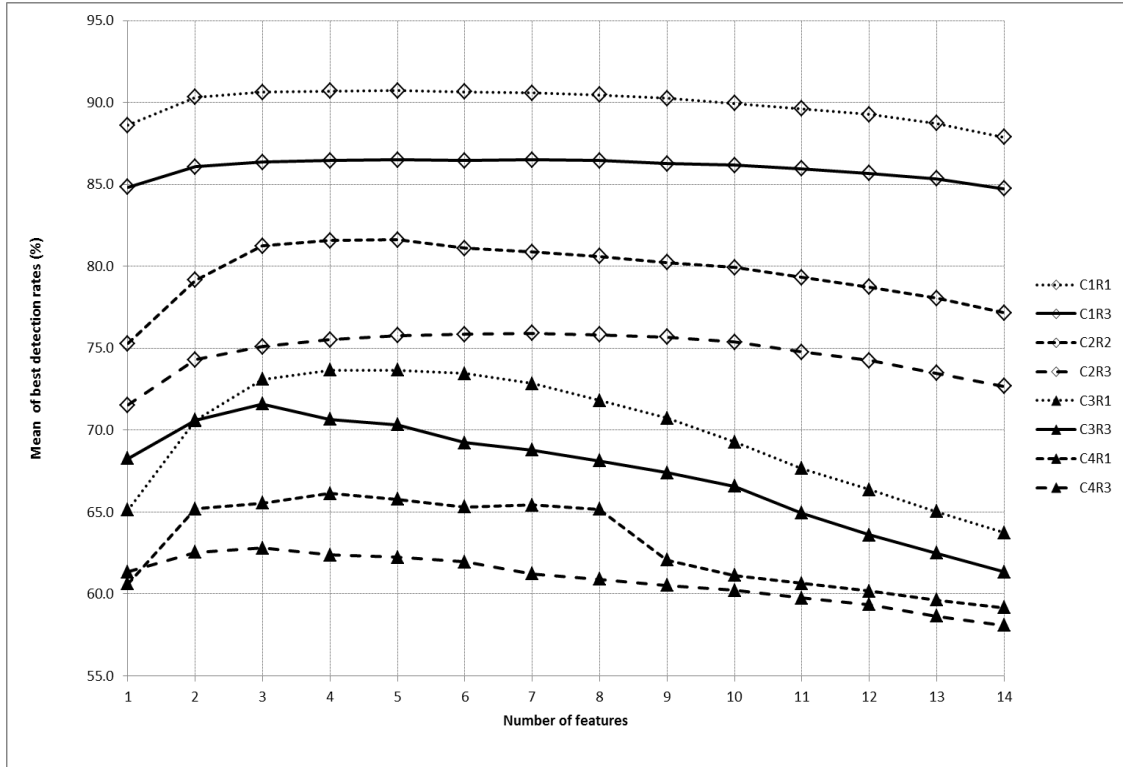


Figure 5-2: Average of best detection rates as function of the number of GLCM features for the different fabric types

Given that we aim at minimising the number of combined features extracted, yet get a good detection rate, we conclude that the optimal number of combined features to use for defect detection is **three**.

5.3 Optimal GLCM feature sets per fabric type

5.3.1 Introduction

From the previous discussion we concluded that the optimal number of GLCM features for fabric defect detection is three. We furthermore need to identify those actual features for each fabric type. Table 5-5 shows the best detection rates as well as the actual features for the different types of fabrics and different types of defects they contain when the number of combined features is three.

Table 5-5: Best triplets of GLCM features for defect detection in different types of fabrics

Type of fabric	Defect type	Best triplets of features and corresponding defect detection rates				GLCM features
		Training set		Testing set		
		Rate	Triplet	Rate	Triplet	
C1R1	E1	94.8	{f2, f10, f12}	94.4	{f8, f10, f14}	{f1}-Angular second moment (Also called energy) {f2}-Contrast {f3}-Correlation {f4}- Sum of squares: variance {f5}-Inverse second moment (also called homogeneity) {f6}-Sum average {f7}-Sum variance {f8}-Sum entropy {f9}-Entropy {f10}-Difference variance {f11}-Difference entropy {f12, f13}- Information measures of correlation {f14}-Maximal correlation coefficient
	E2	92.1	{f8, f11, f14}	91.4	{f5, f11, f12}	
	E3	87.1	{f3, f8, f12}	88.2	{f3, f5, f12}	
	E4	88.9	{f9, f13, f14}	88.2	{f2, f4, f13}	
C1R3	E1	95.0	{f5, f10, f12}	93.9	{f3, f10, f12}	
	E2	83.6	{f7, f13, f14}	84.5	{f10, f11, f13}	
	E3	87.9	{f5, f10, f13}	87.0	{f2, f11, f13}	
	E4	80.0	{f5, f6, f13}	79.0	{f5, f7, f13}	
C2R2	E1	84.0	{f2, f4, f12}	84.1	{f2, f5, f13}	
	E2	85.3	{f4, f5, f12}	87.2	{f5, f7, f12}	
	E3	75.8	{f5, f8, f12}	75.7	{f5, f8, f12}	
	E4	80.2	{f2, f3, f14}	77.6	{f2, f5, f14}	
C2R3	E1	78.0	{f3, f11, f14}	79.5	{f8, f13, f14}	
	E2	76.7	{f5, f7, f12}	77.0	{f5, f6, f12}	
	E3	74.3	{f5, f10, f11}	73.3	{f4, f5, f11}	
	E4	71.4	{f4, f5, f12}	70.7	{f4, f7, f14}	
C3R1	E1	67.7	{f2, f3, f9}	67.3	{f2, f3, f9}	
	E2	80.5	{f1, f2, f13}	79.7	{f2, f4, f14}	
	E3	70.9	{f1, f8, f11}	70.6	{f1, f2, f3}	
	E4	74.4	{f2, f9, f11}	73.7	{f2, f9, f11}	
C3R2	E1	71.9	{f1 ,f11 ,f14}	72.4	{f1 ,f11 ,f14}	
	E2	70.0	{f2 ,f10 ,f14}	68.0	{f4 ,f7 ,f14}	
	E3	73.3	{f9 ,f11 ,f14}	74.7	{f9 ,f11 ,f14}	
	E4	71.5	{f1 ,f9 ,f14}	71.1	{f1 ,f9 ,f14}	
C4R1	E1	66.0	{f1 ,f2 ,f5}	68.9	{f1 ,f2 ,f5}	
	E2	62.6	{f2 ,f3 ,f13}	62.0	{f2 ,f3 ,f13}	
	E3	74.3	{f5 ,f6 ,f11}	73.1	{f5 ,f6 ,f11}	
	E4	59.1	{f6 ,f9 ,f14}	58.5	{f8 ,f11 ,f14}	
C4R3	E1	58.6	{f2 ,f3 ,f11}	58.1	{f2 ,f5 ,f9}	
	E2	73.4	{f6 ,f7 ,f11}	72.0	{f2 ,f4 ,f7}	
	E3	59.0	{f2 ,f3 ,f11}	61.1	{f3 ,f5 ,f11}	
	E4	58.5	{f2 ,f10 ,f11}	61.7	{f5 ,f10 ,f11}	

From Table 5-5 we can read that for a given fabric type the triplet of GLCM features that achieves the best defect detection rate is not necessarily the same for the different types of defects. Furthermore, we can read differences of those best triplets between the training and

the testing set. That means that if one could be sure that a fabric will have only one type of defect, he could optimise the feature extractor to that particular defect by extracting only its corresponding best triplet of GLCM features. However, it is not possible to know beforehand that only one type of defect will occur in a fabric during inspection. Therefore, one should find a triplet of GLCM features that would perform well for all known defects that are likely to occur during inspection.

We propose to select the triplet of GLCM features that minimises the squared error of detection rates calculated from the best detection rates shown in Table 5-5. That squared error would be calculated using the training samples.

5.3.2 The algorithm

For each fabric type, we calculate for each of 364 triplets of GLCM features the defect detection rate for each of the four types of defects. The four obtained detection rates for each defect types are subtracted from their respective possible best rates (as given in Table 5-5) and the resulting differences are individually squared. The four resulting square errors are then added together to get the cumulative squared error (Cum Squared Error) for that triplet of GLCM features and for that fabric type. The diagram of Figure 5-3 illustrates that process for the C1R1 fabric type and the {f1,f2, f3} triplet.

For a given fabric type the cumulative squared error is calculated for each of the 364 triplets of GLCM features and then the triplet that yields the lowest cumulative square error is chosen as the optimal triplet of features for that fabric type. The detailed algorithm is shown in the pseudo-code of Table 5-6.

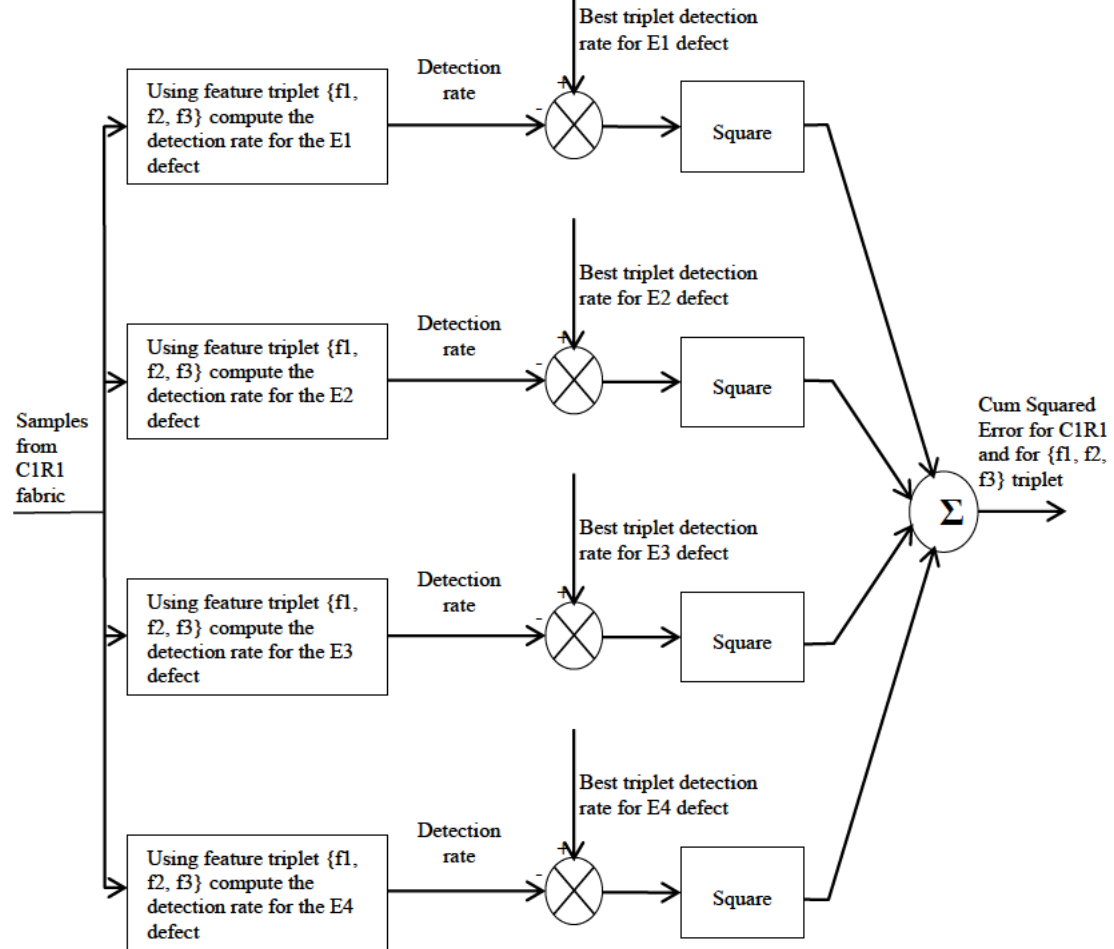


Figure 5-3: Computation of the cumulative square error of the detection rates for the C1R1 fabric type and the {f1, f2, f3} triplet of GLCM features

Table 5-6: Algorithm for identifying the optimal triplet of GLCM features for each fabric type

1)	For each of the eight fabric types (C1R1, C1R3, C2R2, C2R3, C3R1, C3R3, C4R1 and C4R3) samples in the training set
2)	Initialise the Best Cum Squared Error to 40000
3)	Initialise the optimal triplet to {f1, f2, f3}
4)	For each of the 364 possible triplets GLCM features
5)	Initialise the Cum Squared Error to 0
6)	For each of the four defect types (E1, E2, E3 and E4)
7)	Perform the Euclidean distance based detection using the current triplet of the GLCM features.
8)	Calculate the detection rate for the current error type.
9)	Calculate the Squared Error that is equal to the square of the difference of the detection rate calculated in step 6) and the best detection rate of for the current fabric type and the current defect shown in Table 5-5.
10)	Add Squared Error to the previous Cum Squared Error .
11)	Continue with the next defect type
12)	If the current Cum Squared Error is lower than the current Best Cum Squared Error then set Best Cum Squared Error to the current Cum Squared Error and set Optimal triplet to current triplet.
13)	Continue with the next triplet
14)	Record the current optimal triplet as the optimal triplet for the current fabric type.
15)	Continue with the next fabric type

5.3.3 Results and their evaluation

Table 5-7 shows the obtained optimal triplets of GLCM features for each of the fabric types. We can see that the optimal triplet depends on the fabric type as the optimal triplets of the eight fabric types are all different. The most frequent and thus the most discriminative GLCM feature is the contrast {f2} which appears in five optimal triplets out of eight. The sum variance {f7} and the sum entropy {f8} are the least discriminative features as they do not appear in any of the optimal triplets for the eight fabric types used in the experiment.

After finding the optimal triplets of features the next step was to verify whether they possess the desired attribute of being more capable of discriminating defective from defect-free fabric samples than the whole set of the GLCM features. This property is important because the

optimal triplets are limited in size (only three features instead of fourteen) and are therefore much faster to compute than the whole feature set.

Table 5-7: Optimal GLCM features

Fabric type	Optimal feature set	GLCM Features	
C1R1	{f9, f13, f14}	{f1}-Angular second moment (Also called energy)	{f7}-Sum variance
C1R3	{f5, f10, f13}	{f2}-Contrast	{f8}-Sum entropy
C2R2	{f2, f5, f12}	{f3}-Correlation	{f9}-Entropy
C2R3	{f2, f6, f12}	{f4}- Sum of squares: variance	{f10}-Difference variance
C3R1	{f1, f2, f3}	{f5}-Inverse second moment (also called homogeneity)	{f11}-Difference entropy
C3R3	{f9, f11, f14}	{f6}-Sum average	{f12, f13}-Information measures of correlation
C4R1	{f2, f4, f5}		{f14}-Maximal correlation coefficient
C4R3	{f2, f10, f11}		

Table 5-8, Figure 5-4 and Figure 5-5 show the results of the defect detection experiments performed using (i) the best triplet of GLCM features for each particular fabric type and each particular defect type, (ii) the optimal triplet of features for each particular fabric type and (iii) the whole set of fourteen GLCM features.

It can be seen that apart from a few exceptions, the detection rate achieved by the optimal triplet of features lies between that obtained using a triplet optimised for a particular defect and that obtained using the whole set of the fourteen GLCM features. The few exceptions were observed for the samples of the fabric types (i) C1R3 (for the defect type E4 in the training set), (ii) C2R3 (for the defect type E1 in the training set), (iii) C3R1 (for the defect type E2 in the testing set), (iv) C3R3 (for the defect type E2 in the training set) and (v) C4R3 (for the defect type E3 in the testing set). In these cases, the performance achieved by the optimal triplets are close to that obtained using the whole set of features; the highest difference is 2.1%. Therefore the optimal triplets of features do indeed possess the above-mentioned desired attribute.

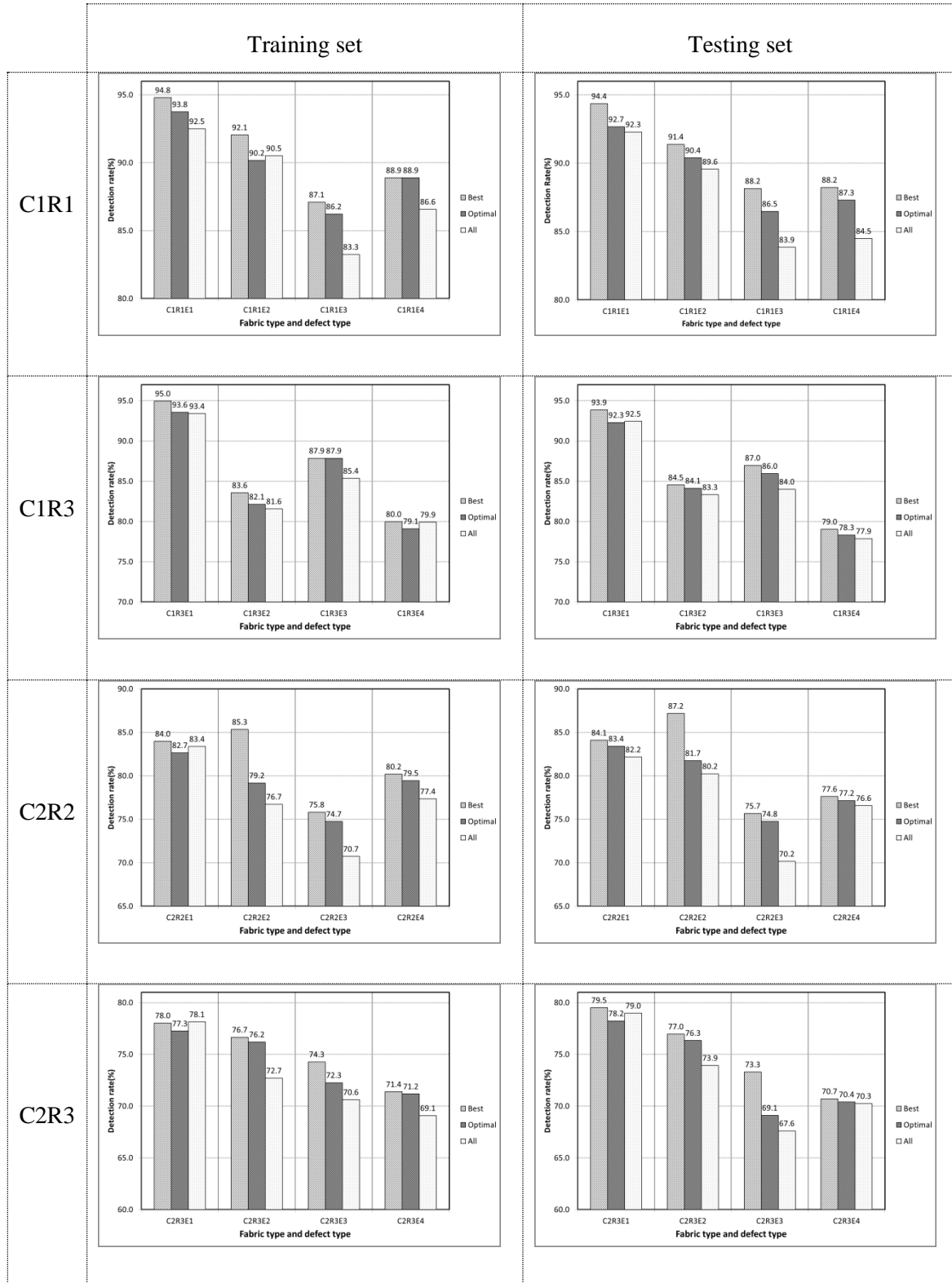


Figure 5-4: Defect detection rate obtained using the best triplet, the optimal triplet as well as the whole set of fourteen GLCM features for the fabric types C1R1, C1R3, C2R2 and C2R3

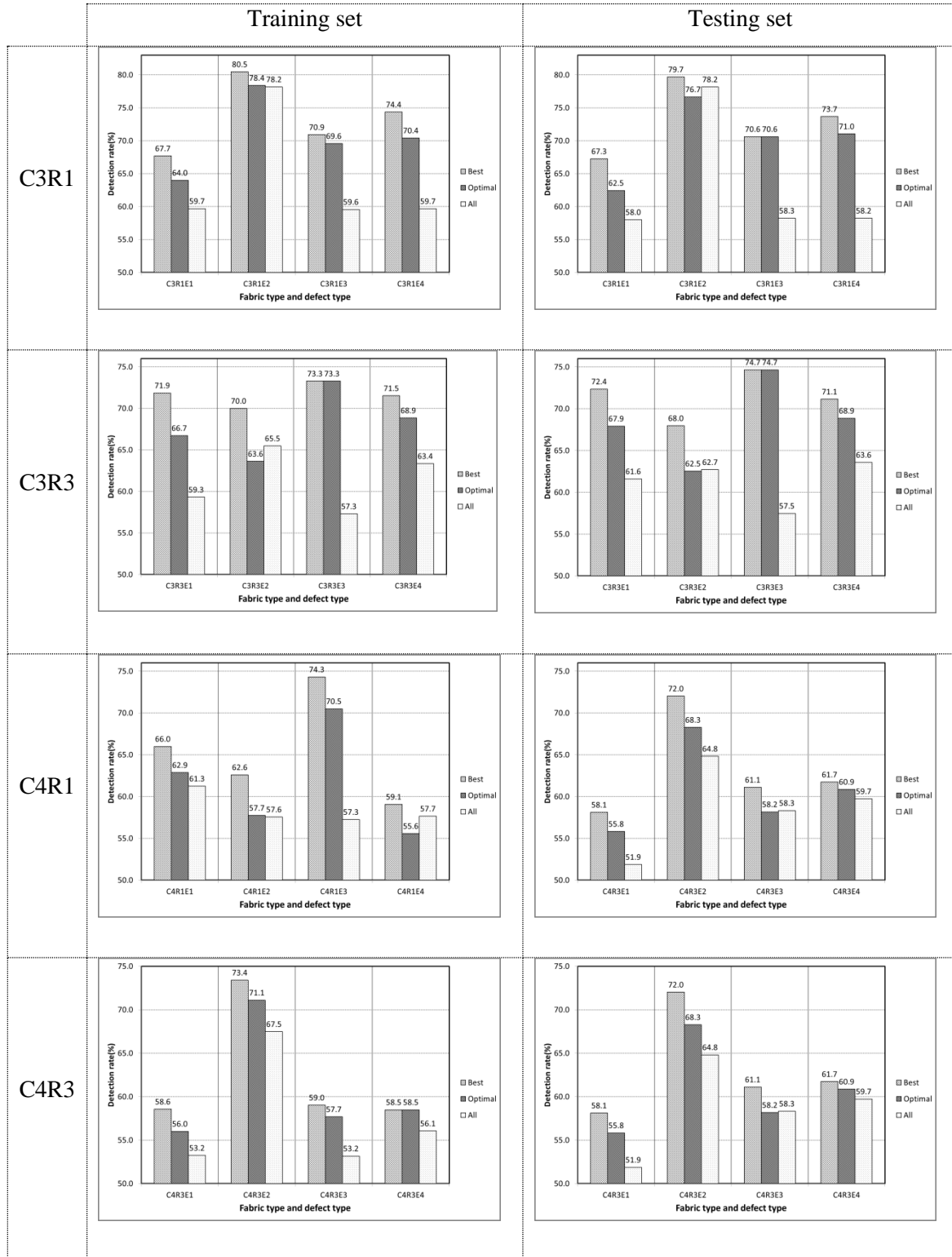


Figure 5-5: Defect detection rate obtained using the best triplet, the optimal triplet as well as the whole set of fourteen GLCM features for the fabric types C3R1, C3R3, C4R1 and C4R4

Table 5-8: Detect performance of the optimal triplet of GLCM features with respect to the best GLCM triplet and the whole GLCM feature set

CRE	Training			Testing		
	Best	Optimal	All	Best	Optimal	All
C1R1E1	94.8	93.8	92.5	94.4	92.7	92.3
C1R1E2	92.1	90.2	90.5	91.4	90.4	89.6
C1R1E3	87.1	86.2	83.3	88.2	86.5	83.9
C1R1E4	88.9	88.9	86.6	88.2	87.3	84.5
C1R3E1	95.0	93.6	93.4	93.9	92.3	92.5
C1R3E2	83.6	82.1	81.6	84.5	84.1	83.3
C1R3E3	87.9	87.9	85.4	87.0	86.0	84.0
C1R3E4	80.0	79.1	79.9	79.0	78.3	77.9
C2R2E1	84.0	82.7	83.4	84.1	83.4	82.2
C2R2E2	85.3	79.2	76.7	87.2	81.7	80.2
C2R2E3	75.8	74.7	70.7	75.7	74.8	70.2
C2R2E4	80.2	79.5	77.4	77.6	77.2	76.6
C2R3E1	78.0	77.3	78.1	79.5	78.2	79.0
C2R3E2	76.7	76.2	72.7	77.0	76.3	73.9
C2R3E3	74.3	72.3	70.6	73.3	69.1	67.6
C2R3E4	71.4	71.2	69.1	70.7	70.4	70.3
C3R1E1	67.7	64.0	59.7	67.3	62.5	58.0
C3R1E2	80.5	78.4	78.2	79.7	76.7	78.2
C3R1E3	70.9	69.6	59.6	70.6	70.6	58.3
C3R1E4	74.4	70.4	59.7	73.7	71.0	58.2
C3R3E1	71.9	66.7	59.3	72.4	67.9	61.6
C3R3E2	70.0	63.6	65.5	68.0	62.5	62.7
C3R3E3	73.3	73.3	57.3	74.7	74.7	57.5
C3R3E4	71.5	68.9	63.4	71.1	68.9	63.6
C4R1E1	66.0	62.9	61.3	68.9	66.9	65.3
C4R1E2	62.6	57.7	57.6	62.0	58.2	57.3
C4R1E3	74.3	70.5	57.3	73.1	71.3	60.3
C4R1E4	59.1	55.6	57.7	58.5	55.2	56.8
C4R3E1	58.6	56.0	53.2	58.1	55.8	51.9
C4R3E2	73.4	71.1	67.5	72.0	68.3	64.8
C4R3E3	59.0	57.7	53.2	61.1	58.2	58.3
C4R3E4	58.5	58.5	56.1	61.7	60.9	59.7

5.4 Performance of the GLCM features as function of the quantisation level of the image

5.4.1 Introduction

If G is the number of grey levels (quantisation level) of an image then the size of its grey level co-occurrence matrix is G^2 . The computational cost to compile such a matrix and to extract

features from it increases rapidly with G . It is therefore important to decrease the number of grey levels of an image before compiling its grey level co-occurrence features in order to shorten the time required for feature extraction. However, such a reduction of the number of grey levels should not compromise the defect detection ability of the resulting features.

The aim of the experiments under this section is to find the relationship between the number of grey levels of fabric images and the rate of detection of defects from them. That would allow making a suitable choice of the number of grey levels to be used for fast extraction of GLCM co-occurrence features without compromising the defect detection performance.

5.4.2 Description of the experiment

The experiment was performed using the optimal triplets of GLCM features for each fabric type as found in Section 5.3. The steps of the experiment were as follows:

1. Quantise the grey levels of all images in the dataset from which the experimental samples were extracted using 16, 32, 64, 128 and 256 grey levels. The uniform quantisation algorithm [47] was used to that effect.

Let f be the original image, f_{max} be the maximal value of f , f_{min} be the minimum value of f , G be the number of desired grey levels. The quantised image g is given by (5.3).

$$g = \frac{f - f_{min}}{f_{max} - f_{min}} (G - 1) \quad (5.3)$$

2. For each quantisation level G , compute the optimal triplets of GLCM features for all the samples from both the training and the testing set grouped by fabric type.
3. For each quantisation level G , and for each fabric type, train a Euclidean distance classifier using the ML algorithm and perform the defect detection experiment for both the training and the testing sets.
4. Record the defect detection rates and study for every fabric type their variation as the quantisation level G changes.

5.4.3 Results and interpretation

Table 5-9 and Table 5-10 show the variation of the defect detection rate as the quantisation level changes for each type of fabric, while Figure 5-6 and Figure 5-7 show the same information graphically.

Table 5-9: Detection rate (%) vs quantisation level for the training set

Fabric type	Quantisation level				
	16	32	64	128	256
C1R1	87.5	87.3	87.8	88.4	88.4
C1R3	83.5	84.3	84.8	85.7	86.0
C2R2	76.9	78.2	77.0	74.9	75.3
C2R3	75.5	76.1	76.0	75.9	75.4
C3R1	66.9	68.1	68.1	67.9	67.9
C3R3	65.1	64.2	63.1	61.6	61.9
C4R1	59.6	60.2	60.0	60.1	60.0
C4R3	59.4	59.6	59.8	59.8	59.8

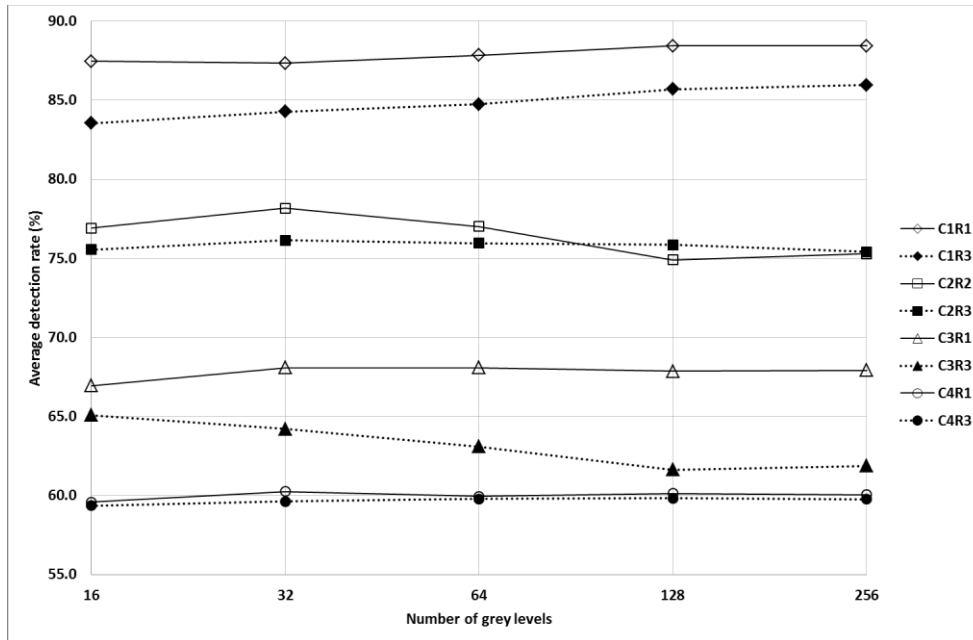


Figure 5-6: Detection rate as function of the quantisation level for the samples in the training set and for different types of fabrics

The results show that there is little variation of the average detection rate when the quantisation level changes. The highest variation is observed for the C3R3 fabric type, where the average detection rate shows a decrease of 3.9% (for the testing set) when the quantisation level changes from 16 to 256. The results show a similar order of magnitude for the variation of the average detection rate for C1R3 and C2R2. For all the other fabric types, the variation is under 1.2%. According to the results, a safe choice of the quantisation level is $G=64$.

The same trend of variation in the detection rate is observed for both the training and the testing set. That means that the observations and conclusions made based on the training samples can be extended to unknown samples.

Table 5-10: Detection rate (%) vs quantisation level for the testing set

Fabric type	Number of grey levels				
	16	32	64	128	256
C1R1	87.8	87.8	88.7	89.0	88.4
C1R3	82.9	83.9	84.3	85.3	85.3
C2R2	75.9	77.9	77.9	75.5	75.8
C2R3	76.1	76.0	75.8	75.7	75.6
C3R1	66.1	66.5	66.7	66.5	66.6
C3R3	65.5	65.2	63.2	61.8	61.6
C4R1	60.8	60.4	60.9	60.4	60.1
C4R3	59.2	59.8	59.9	60.1	60.3

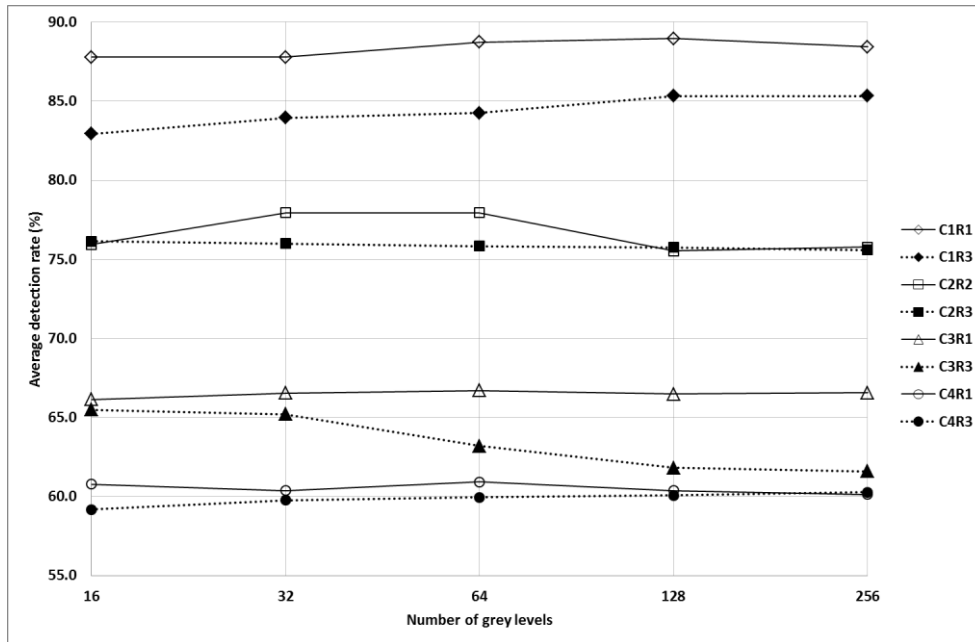


Figure 5-7: Detection rate as function of the quantisation level for the samples in the testing set and for different types of fabrics

5.5 Performance of the GLCM features as function of the interpixel distance

5.5.1 Introduction

The interpixel distance d used to compile the GLCM matrix is an important parameter. In literature its choice is not always motivated and $d=1$ is usually selected [18, 21, 31, 37, 38, 44].

The aim of the experiment under this section is to find out whether that choice is optimal or whether a better choice could be justified.

5.5.2 Description of the experiment

The experiment is performed using optimal feature sets as found in Section 5.3, using 64 grey levels as discussed in Section 5.4. The steps of the experiment were as follows:

1. Extract the experimental samples from all the fabric images quantised using the 64 grey levels. The experimental samples are exactly those described in Section 4.5 (the same samples used in all the experiments of this thesis).
2. Compile the GLCMs of each of the samples from both the training and the testing sets using the interpixel distance $d=1$, then using the interpixel distance $d=2$ and so forth up to the interpixel distance $d=10$.
3. For each value of the interpixel distance parameter d , and for each fabric type, extract the optimal GLCM features that make up the optimal triplet of features for that particular fabric type. Then perform the defect detection experiment for both the training and the testing sets using the obtained features.
4. Record the defect detection rates and study for every fabric type their variation as the interpixel distance value d varies from 1 to 10.

5.5.3 Results and interpretation

Table 5-11 and Table 5-12 show the variation of the defect detection rate as the quantisation level changes for each type of fabric, while Figure 5-8 and Figure 5-9 show the same information graphically.

The results show that the defect detection rate decreases when the interpixel distance d is increased and that the best detection rates are recorded for $d=1$. The only exception is observed for the fabric type C3R3, which shows a slight increase in detection as the interpixel distance increases from 1 to 4 and then a slight decrease as d is further increased from 4 to 10. This is probably due the periodic nature of the pattern on the fabric of type C3R3 as shown in Figure 4-2.

These considerations are consistent with the choice of interpixel $d=1$ adopted by numerous researchers and we therefore recommend it for fabric defect detection.

Table 5-11: Detection rate (%) vs interpixel distance for the training set

Fabric type	Interpixel distance (d)									
	1	2	3	4	5	6	7	8	9	10
C1R1	87.8	87.9	87.7	87.5	87.1	87.0	86.8	87.5	86.9	86.8
C1R3	84.8	83.5	81.3	79.5	78.9	77.8	77.4	77.0	77.2	77.2
C2R2	77.0	76.2	75.0	72.1	67.9	65.9	64.9	63.4	63.3	63.3
C2R3	76.0	74.9	73.1	72.2	70.0	68.7	67.4	66.6	66.1	65.9
C3R1	68.1	66.1	64.3	64.5	62.9	63.4	64.7	64.5	64.2	65.6
C3R3	63.1	63.8	65.6	66.2	66.0	64.8	65.2	64.6	64.6	63.6
C4R1	60.0	58.0	57.8	57.1	57.4	57.2	57.8	58.0	58.0	58.1
C4R3	59.8	57.7	56.3	56.7	57.8	57.8	58.3	58.5	58.0	58.2

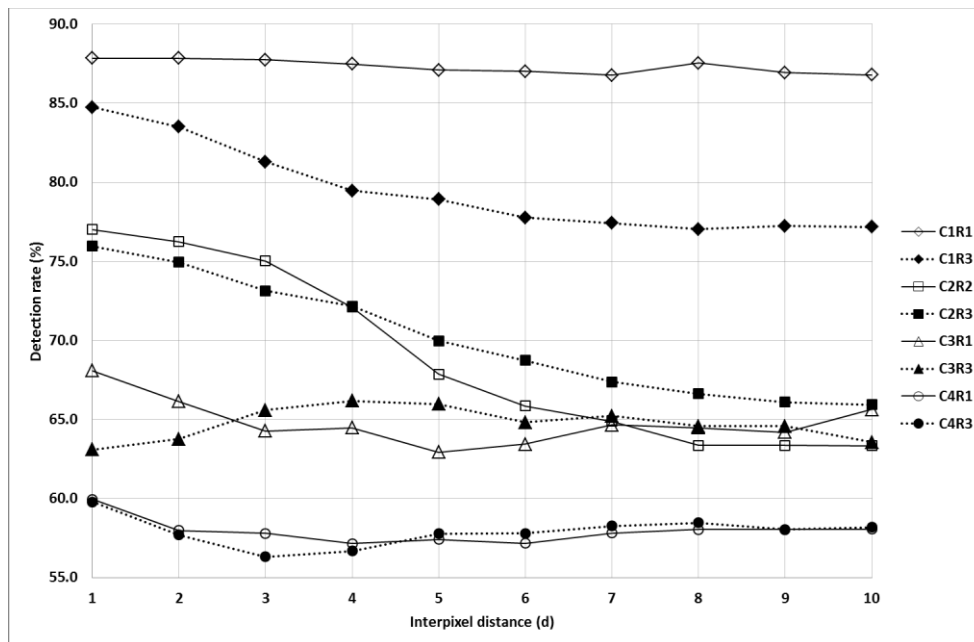


Figure 5-8: Detection rate as function of the interpixel distance for the samples in the training set and for different types of fabrics

Table 5-12: Detection rate (%) vs interpixel distance for the testing set

Fabric type	Interpixel distance (d)									
	1	2	3	4	5	6	7	8	9	10
C1R1	88.7	88.4	88.2	87.4	87.6	88.1	87.9	87.8	87.9	87.6
C1R3	84.3	83.2	81.9	79.4	79.5	78.1	77.9	78.1	77.5	78.1
C2R2	77.9	76.8	75.4	72.1	67.8	65.9	64.5	63.4	63.3	63.2
C2R3	75.8	74.9	72.9	71.8	70.2	67.9	66.6	65.9	65.5	65.3
C3R1	66.7	65.3	63.6	64.1	63.5	64.4	64.8	64.5	63.7	65.4
C3R3	63.2	64.3	65.1	66.4	66.4	64.2	63.9	63.9	64.4	64.1
C4R1	60.9	59.3	59.0	58.4	58.0	57.6	58.3	57.0	56.9	57.4
C4R3	59.9	57.9	56.7	56.8	57.6	58.6	59.0	59.0	59.0	58.9

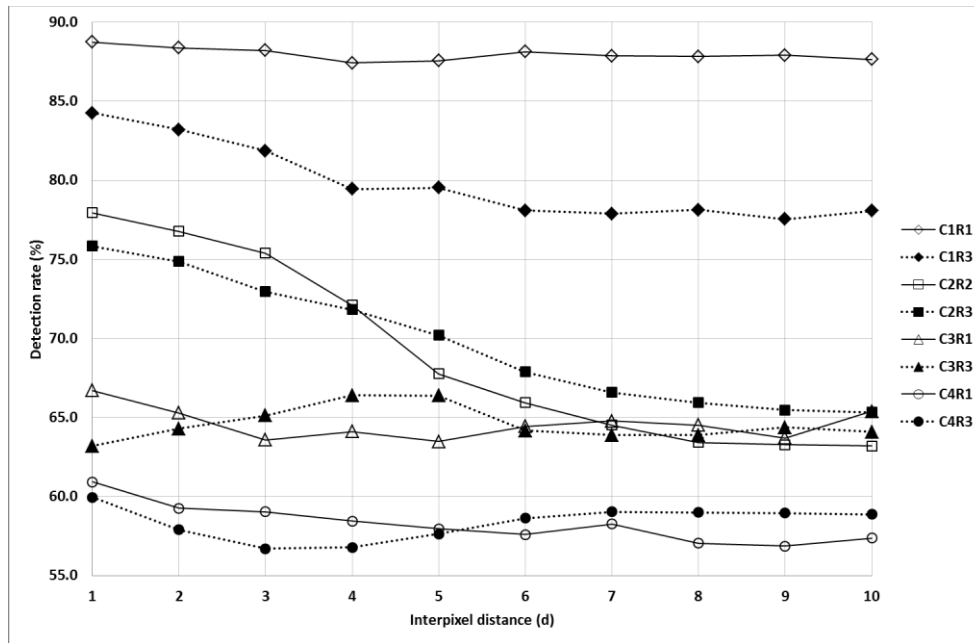


Figure 5-9: Detection rate as function of the interpixel distance for the samples in the testing set and for different types of fabrics

5.6 Effect of the classifier on the performance of GLCM features

5.6.1 Introduction

We compare the performance of the GLCM features when used for fabric defect detection using (i) the Euclidean distance classifier trained using the maximum likelihood (ML) method, (ii) the Euclidean distance classifier trained using the minimum classification error (MCE), (iii) the K-nearest-neighbour classifier (K=3) and (iv) the feed-forward neural network distance classifier.

We first check the validity of the observations made under sections 5.2 and 5.3 regarding the optimal number of GLCM features, as well as the corresponding optimal triplets of features when classifiers other than the ML-trained Euclidean distance classifier are used. We then study the detection rates obtained when the different above-mentioned classifiers are used. Finally, we study the generalisation ability of those classifiers by comparing the difference of the detection rates between the samples of the training set and those of the testing set. A large difference indicates poor generalisation ability.

5.6.2 Dependence of the optimal feature set on the classifier

In sections 5.2 and 5.3 we concluded that the number of optimal GLCM features is three and that the specific features vary from one fabric type to the other. However, those observations were made using the ML-trained Euclidean distance classifier. We want to find out whether those conclusions still hold true when other classifiers are used. Therefore we perform the same experiments, but using (i) the MCE-trained Euclidean distance classifier, (ii) the K-nearest-neighbour classifier (K=3) and (iii) the feed-forward neural network classifier.

The characteristics of the neural network used were as follows: The number of inputs was twice the number of GLCM features because each GLCM was made up of two sub-features (the mean and the range). As explained in Subsection 3.5.5, we used one hidden layer where the number of neurons was given by (5.4)

$$N_{hidden} = fix\left(\frac{2}{3} \times N_{inputs}\right) + 2 \quad (5.4)$$

where N_{hidden} is the number of neurons in the hidden layer, N_{inputs} is the number of inputs to the neural network while the $fix(.)$ function represents the whole part of a positive real number. The output layer had two neurons (Neuron1 and Neuron2) to represent either the defect-free or defective state of the sample whose features were inputs to the neural network. The coding of the output neurons was as follows: Neuron1=0 and Neuron2=1 meant that the sample was defect-free, while Neuron1=1 and Neuron2=0 meant that the sample was defective.

The feed-forward neuron network classifier was implemented using the MATLAB Neural Network Toolbox. The following training parameters were used:

- Training strategy: Early stopping
- Training function: Levenberg-Marquardt backpropagation (trainlm)
- Maximum number of training epochs: 1000

To prevent the problem of undertraining that might result from the high training speed of the Levenberg-Marquardt algorithm, the training of the network was repeated after reinitialisation every time that the final mean square error was higher than 0.1. After twenty repeats with no mean square error lower than 0.1, the trained network with the lowest mean square error among the twenty was used.

For each of the above-mentioned classifiers we performed the defect detection experiments using subsets of one, two, three and so forth up to fourteen GLCM features. For each classifier and for each cardinality of features we recorded the subset that yielded the highest detection rate as well as the obtained detection rate. For each cardinality of feature subsets (one through fourteen), the average of best detection rates across fabric types was calculated. Table 5-13 and Figure 5-10 show the results.

We see that the same trend observed for the ML-trained Euclidean distance classifier (as shown in Figure 5-1) of an increase in detection rate with the increasing number of GLCM features, followed by a drop remains valid. However, we observe specific differences for each classifier.

For the MCE-trained Euclidean distance classifier, we observe a substantial rise of the detection rate when the number of features is increased from one to two and then a lower rise when the number of features is increased from two to three. A further lower improvement is observed then the number of features goes from three to four and an even lower improvement when the number of features goes from four to five. There is negligible improvement with the change of number of features from five to eight, and then a drop of performance when the number of features is increased beyond eight. The drop of the detection rate is small in magnitude so that there is no big penalty in detection performance when we use the full set of fourteen GLCM features. However, using the full set of features would lead to unnecessary increase of computational cost. Therefore, considering both the defect detection performance and the computational cost, the optimal number of features to use when a MCE-trained Euclidean distance classifier is used is **five**.

For the K-nearest-neighbour classifier (K=3) we observe from Figure 5-10 almost the same trend of change in the detection rate with respect to the number of GLCM features as the MCE-trained Euclidean distance classifier. However the detection rate for the K-nearest-classifier is higher by an absolute amount of about 3%. In addition, the decrease of the detection rate beyond eight features is a slightly more pronounced for the K-nearest-neighbour classifier than for the MCE-trained Euclidean distance classifier. The optimal number of GLCM features to be used for the K-nearest-neighbour classifier is also **five**.

For the feed-forward neural network classifier, Figure 5-10 shows that the best detection rate continues to rise with the number of GLCM features until the maximum of nine features is reached. Beyond nine features, the best detection rate slowly drops as the number of features increases. Therefore, the optimal number of GLCM features when a feed-forward neural network classifier is used is **nine**.

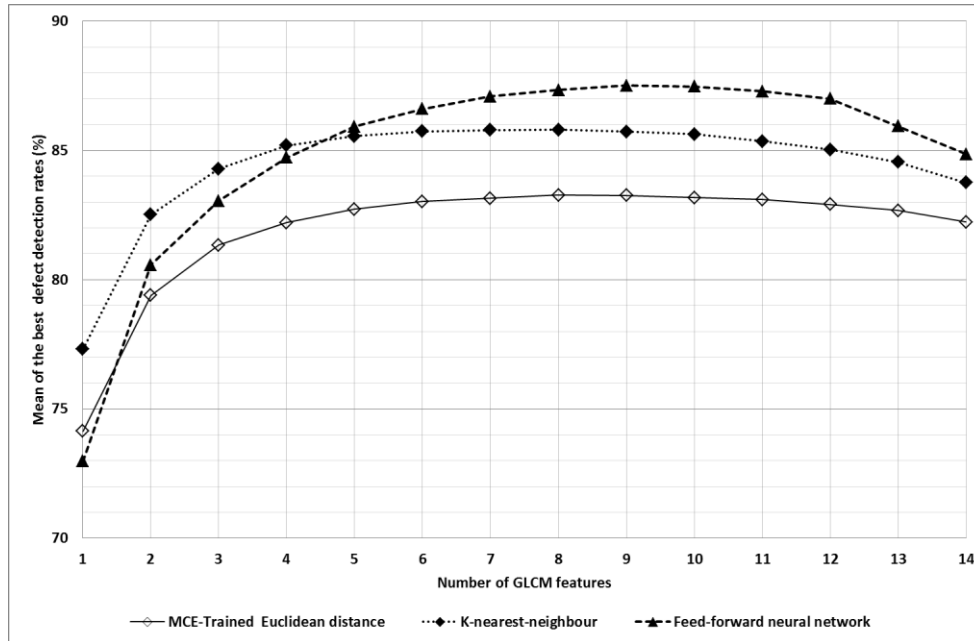


Figure 5-10: Mean of the best detection rates as function of the number of GLCM features for all the samples in the dataset when the MCE-trained Euclidean distance classifier, K-nearest-neighbour classifier and feed-forward neural network classifier are used

Table 5-13: Mean of the best detection rates as function of the number of GLCM features for different classifiers

Classifier	Number of GLCM features													
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
MCE-trained Euclidean distance	74.1	79.4	81.3	82.2	82.7	83.0	83.1	83.3	83.3	83.2	83.1	82.9	82.7	82.2
K-nearest-neighbour	77.3	82.5	84.3	85.2	85.6	85.7	85.8	85.8	85.7	85.6	85.4	85.0	84.5	83.8
Feed-forward neural network	73.0	80.6	83.0	84.7	85.9	86.6	87.1	87.3	87.5	87.5	87.3	87.0	85.9	84.9

Looking at the graph in Figure 5-11, showing the mean best detection rates versus the number of features for individual fabric types when the MCE-trained Euclidean distance classifier is

used, we can see that the conclusion of five optimal GLCM features holds for individual fabric types.

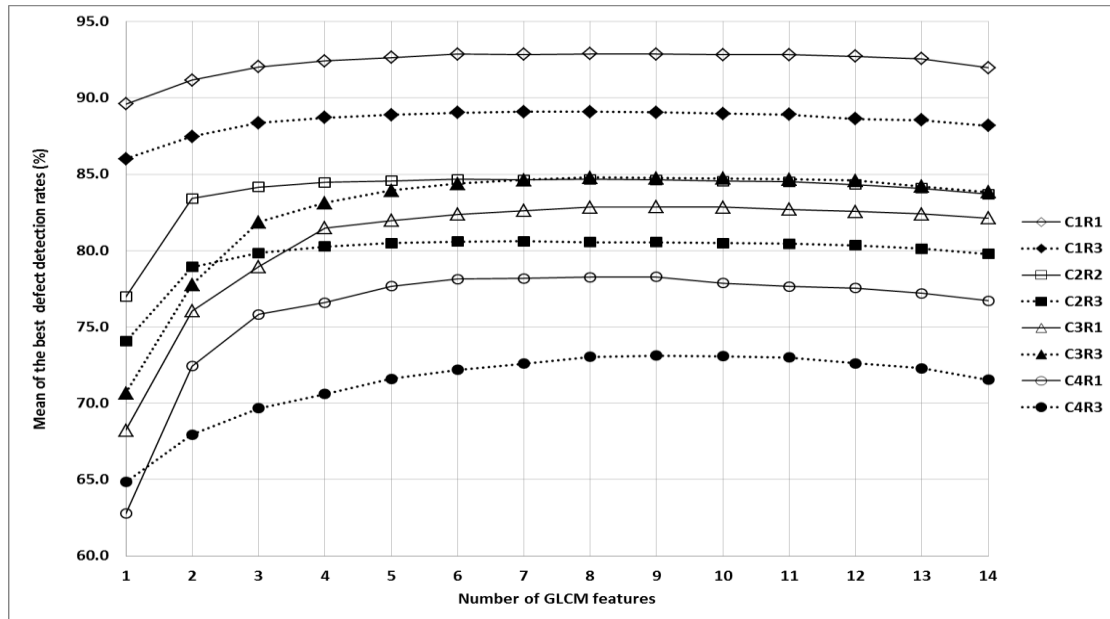


Figure 5-11: Mean best detection rates as function of the number of GLCM features for the samples from the different types of fabric when a MCE-trained Euclidean distance classifier is used.

The conclusion that there are five optimal features when a K-nearest-neighbour classifier holds also for individual fabric types as it can be seen from Figure 5-12.

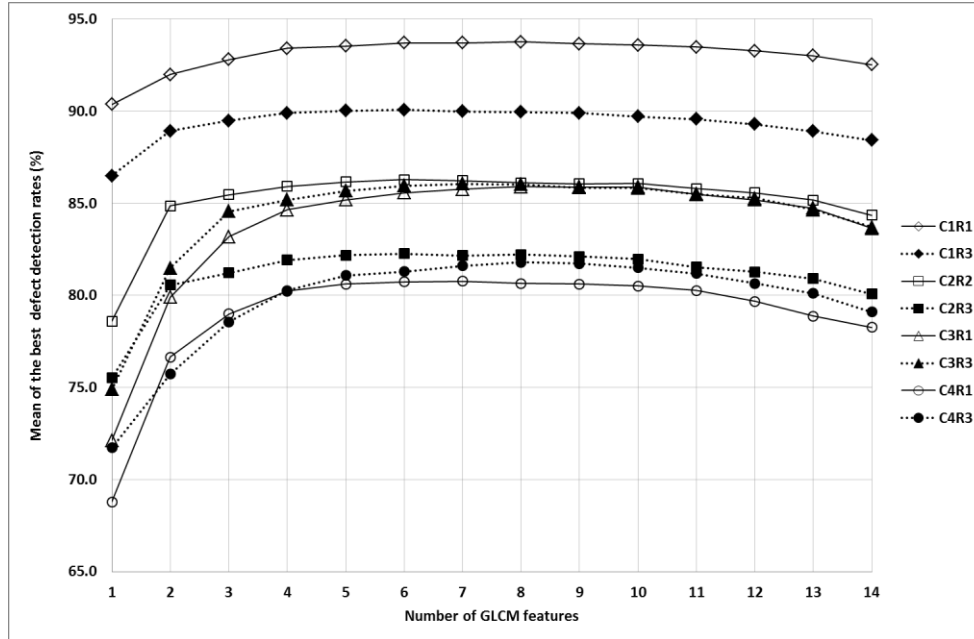


Figure 5-12: Mean best detection rates as function of the number of GLCM features for the samples from the different types of fabric when a K-nearest-neighbour classifier is used.

Figure 5-13 shows that similarly the observation of nine optimal GLCM features holds when a feed-forward neural network classifier is used.

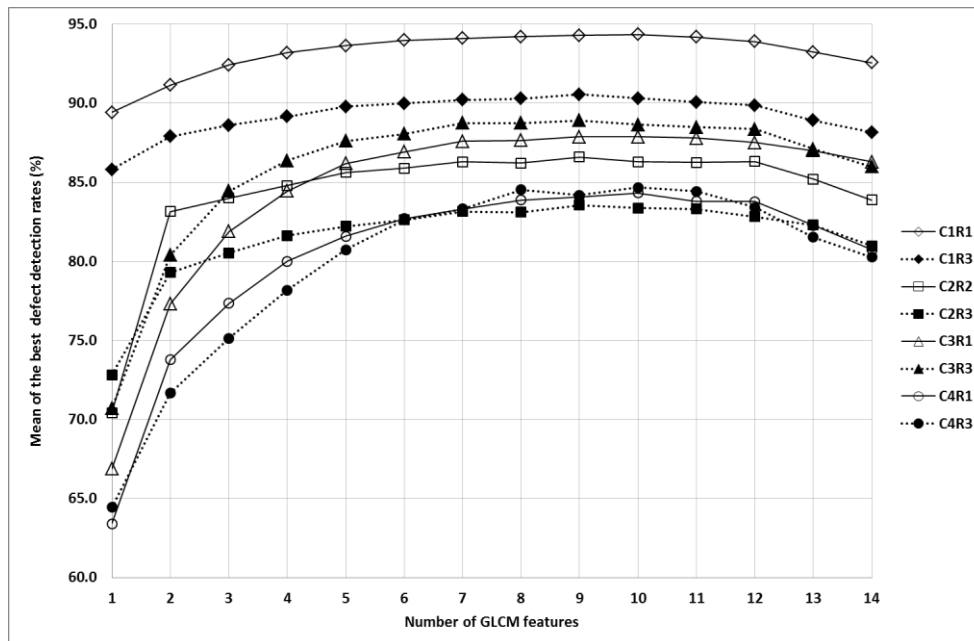


Figure 5-13: Mean best detection rates as function of the number of GLCM features for the samples from the different types of fabric when a feed-forward neural network classifier is used.

In short, the number of optimal GLCM features depends on the classifier used and the results seem to indicate that the better the detection accuracy of the classifier, the more capable it is of using more features for improved detection rate. Table 5-14 summarises these results.

Table 5-14: Number of optimal GLCM features and average of the best detection rates for different classifiers

Classifier	Number of optimal GLCM features	Average best detection rate (%)
Euclidean (ML training)	3	75.8
Euclidean (MCE training)	5	82.7
K-nearest-neighbour (K=3)	5	85.5
Feed-forward neural network	9	87.5

After determining the number of optimal GLCM features when the MCE-trained Euclidian distance, K-nearest-neighbour and feed-forward neural network classifiers are used, we use the algorithm in Subsection 5.3.2 to identify the actual optimal GLCM feature set for each of those classifiers and for each of eight fabric types in our dataset. Table 5-15 shows the results.

Table 5-15: Optimal GLCM feature sets for different classifiers for the eight fabric types

Classifier	Euclidean (ML training) Optimal triplets	Euclidean (MCE training) Optimal quintuplets	K-nearest-neighbour Optimal quintuplets	Feed-forward neural network Optimal nonuplets
C1R1	{f9, f13, f14}	{f6, f10, f11, f13, f14}	{f2, f6, f10, f11, f14}	{f2, f5, f6, f7, f9, f10, f11, f13, f14}
C1R3	{f5, f10, f13}	{f2, f5, f7, f13, f14}	{f5, f7, f8, f10, f11}	{f1, f2, f5, f7, f8, f9, f10, f11, f14}
C2R2	{f2, f5, f12}	{f5, f8, f9, f11, f13}	{f2, f6, f8, f11, f14}	{f1, f2, f3, f4, f5, f10, f11, f12, f14}
C2R3	{f2, f6, f12}	{f2, f4, f11, f12, f14}	{f1, f2, f11, f12, f13}	{f2, f4, f5, f7, f9, f10, f11, f12, f14}
C3R1	{f1, f2, f3}	{f2, f6, f9, f11, f14}	{f2, f4, f9, f11, f13}	{f1, f2, f6, f8, f9, f10, f11, f12, f14}
C3R3	{f9, f11, f14}	{f2, f5, f9, f10, f14}	{f1, f7, f8, f11, f14}	{f2, f3, f4, f5, f6, f8, f10, f12, f13}
C4R1	{f2, f4, f5}	{f5, f10, f11, f13, f14}	{f4, f8, f9, f11, f14}	{f3, f5, f6, f7, f8, f9, f10, f11, f14}
C4R3	{f2, f10, f11}	{f2, f3, f7, f8, f10}	{f1, f6, f10, f11, f14}	{f2, f4, f6, f7, f9, f10, f11, f12, f14}
GLCM features		<div> <div> {f1}-Angular second moment (Also called energy) {f2}-Contrast {f3}-Correlation {f4}- Sum of squares: variance {f5}-Inverse second moment (also called homogeneity) {f6}-Sum average </div> <div> {f7}-Sum variance {f8}-Sum entropy {f9}-Entropy {f10}-Difference variance {f11}-Difference entropy {f12, f13}-Information measures of correlation {f14}-Maximal correlation coefficient </div> </div>		

5.6.3 Defect detection performance for different classifiers

Under this subsection, we perform defect detection for all the samples in the dataset using the MCE-trained Euclidean distance, K-nearest-neighbour and feed-forward neural network classifiers. We compare the performance obtained using all the features in the dataset to that obtained using only the optimal subsets identified in Subsection 5.6.2. The experiments steps were as follows for each of the classifiers:

1. GLCM feature extraction for all the samples in the dataset
2. Normalisation of the features of the training samples for each fabric type and for each defect type. This normalisation step should also provide the normalisation parameters to be used for the samples of the corresponding testing set. This step varies from one classifier to the other.

3. Training of the classifier using the normalised GLCM features of the training set obtained from step 2. Training will be different for the different classifiers.
4. Classification of each sample of the training set as defective or defect-free by feeding its normalised features into the trained classifier. Calculate the detection rate by dividing the number of the samples correctly classified by the total number of training samples of that fabric type with that particular defect type.
5. Normalisation of the features of the testing samples for each fabric type and each defect type using the normalisation parameters obtained from step 2.
6. Classification of each sample of the testing set as defective or defect-free by feeding its normalised features into the trained classifier. Calculate the detection rate by dividing the number of the samples correctly classified by the total number of testing samples of that fabric type with that particular defect type.

All the steps are similar for all the classifiers apart from the step of feature normalisation, classifier training and feature classification that may vary from one classifier to the other.

The type of feature normalisation to use will depend on the kind of classifier used. For the Euclidean distance classifiers, whether trained using the ML or MCE algorithms, and the K-nearest-neighbour classifier, we will use the min-max normalisation as given by (3.118). For the feed-forward neural network classifier, we will use the min-max normalisation that linearly scales the input data into the range $[-1, 1]$ rather than $[0, 1]$ using the MATLAB function '*mapminmax*'. That is because the sigmoid transfer functions used in our neural networks have input ranges that are symmetric with respect to 0.

Classifier training also varies from one type of classifier to the other as described in Section 3.5. For the ML-trained Euclidean distance classifier, training is simply calculating the mean feature vector of each feature class of the training set. For the MCE-trained Euclidean distance classifier, the mean feature vectors of each feature class are changed adaptively to get a reference feature vectors (one for each class) that minimises the classification rate of the samples in the training set. We implemented the related optimisation process using the MATLAB optimisation function '*fminunc*'. The K-nearest-neighbour classifier is trained by simply storing all the feature vectors of each class in the training set. The feed-forward neural network classifier is trained using the backpropagation algorithm as described in Subsection 3.5.5.

The results are shown in figures 5-14 through 5-16.

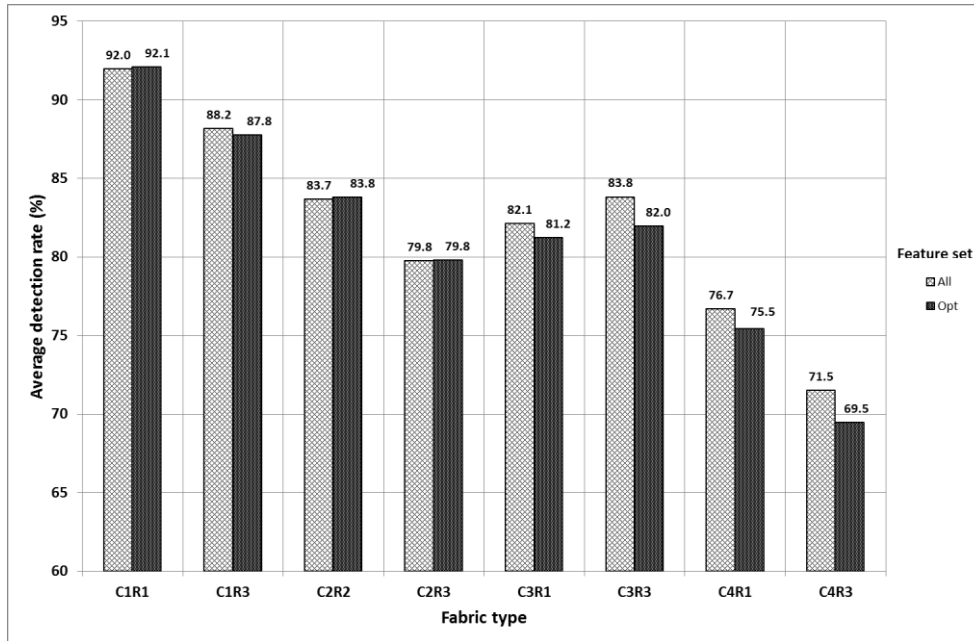


Figure 5-14: Compared detection rate of the optimal feature set to the full set (14 features) when the MCE-trained Euclidean distance classifier is used

From Figure 5-14 we see that for the MCE-trained Euclidean distance classifier, the detection rate of the optimal feature set is almost the same as that obtained using the full feature set for fine fabrics (C1R1, C1R3, C2R2 and C2R3), while for the remaining fabric types, the detection rate obtained using the full feature set is slightly higher than obtained using the optimal feature set. This is acceptable, given that the difference is small (the highest difference is of 2%), and because the final aim is to combine those features with others (wavelet and MRF-based). We prefer the optimal set because it contains fewer features (five instead of fourteen) and therefore takes much less time to extract.

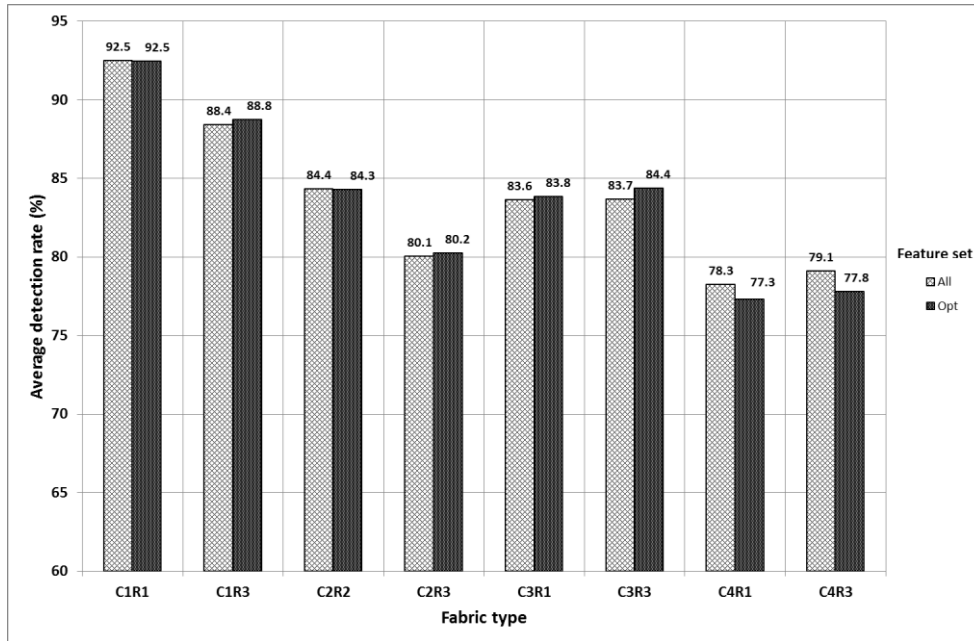


Figure 5-15: Compared detection rate of the optimal feature set to the full set (fourteen features) when the K-nearest-neighbour classifier is used

Figure 5-15 shows that using the K-nearest-neighbour classifier yields better detection rates of the optimal set than the full set for most of the fabric types with the exception of the fabrics with no apparent periodicity (C4R1 and C4R3). Even in that latter case, the difference of average detection rates is small (1% for C4R1 and 1.3% for C4R3).

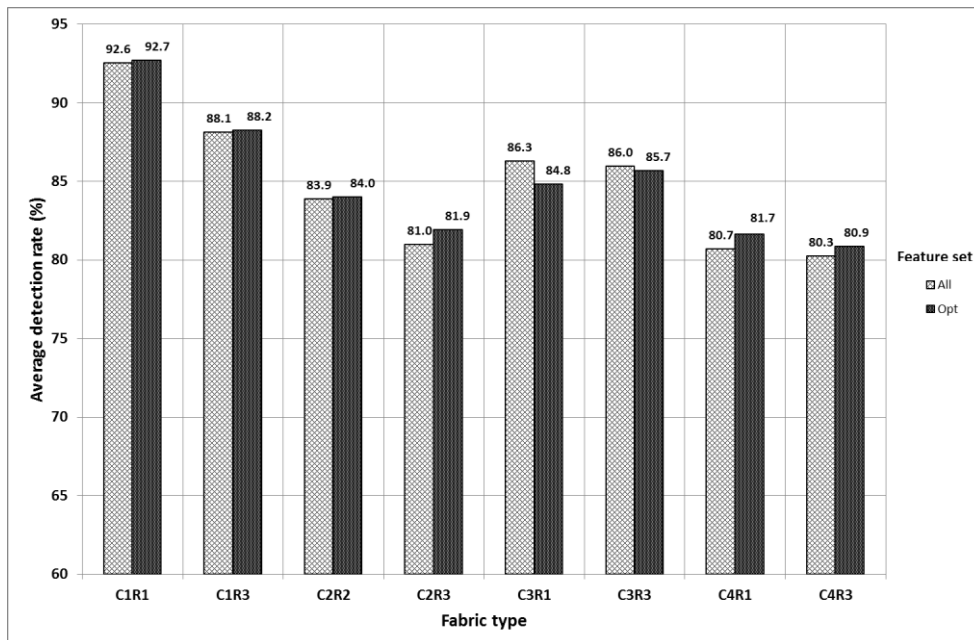


Figure 5-16: Compared detection rate of the optimal feature set to the full set (fourteen features) when the feed-forward neural network classifier is used

Figure 5-16 shows that using the feed-forward neural network classifier, the optimal feature set yields higher detection rates than those yielded by the full feature set for most of the fabric types. The two exceptions are the fabrics with a visible periodic structure (C3R1 and C3R3). In these cases, the difference of average detection rates between the full set and the optimal set is small (1.5% for C3R1 and 0.3% for C3R3).

In conclusion, the use of the optimal GLCM feature set instead of the full feature set is advantageous because it allows for better, same or slightly lower defect detection rates while taking much less time to extract given their reduced number.

5.6.4 Generalisation ability of classifiers

Figure 5-17 compares the average detection rate for the training and testing set of different classifiers. We can see that the MCE-trained Euclidean distance classifier offers the best generalisation ability among the classifiers we used as the difference of average detection rates between the training and the testing set is the smallest (1.9%).

The next best generalisation ability is obtained with ML-trained Euclidean distance classifier (difference=3.1%) followed closely by the feed-forward neural network classifier (difference=3.7%). The K-nearest-neighbour classifier offers the worst generalisation ability (difference=10.4%), because any sample of the training set has itself as its nearest neighbour and therefore has a high probability of being correctly classified. This tendency increases when the number K of neighbours used by the classifier decreases. We are using K=3.

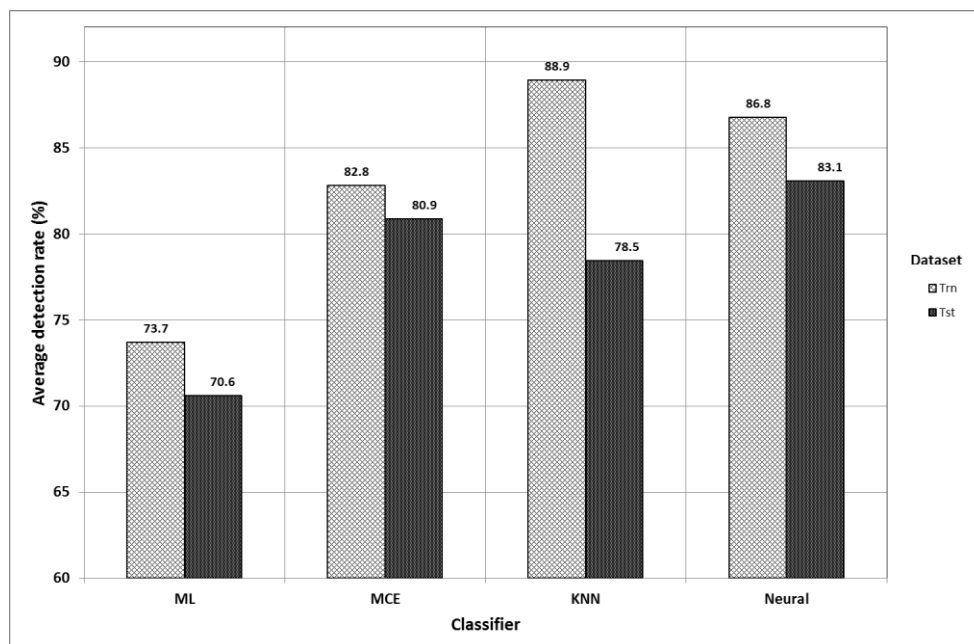


Figure 5-17: Generalisation ability of different classifiers

5.7 Defect classification using GLCM features

5.7.1 Introduction

In this section, we classify the experimental samples into defect classes using GLCM features. Five defect classes are available in the dataset for every fabric type; (i) E0: the defect-free class, (ii) E1: holes in the fabric and cuts caused by mechanical damage, (iii) E2: oil stain and colour fading, (iv) E3: thread errors: condensations of filaments (not mechanically induced cracks), absence of individual threads in the fabric and (v) E4: foreign body in the fabric.

We perform the defect classification in two stages: (i) defect detection and (ii) defect classification. For each fabric type, the first stage consists in separating defective samples from defect-free samples. The second stage deals with samples classified as defective and assign them to one of the four defect classes (the defect-free class is not included).

We compare the performance of the optimal sets of features with respect to the full set of GLCM features using four classifiers: (i) The Euclidean distance classifier trained using the ML algorithm, (ii) the Euclidean distance classifier trained using the MCE algorithm, (iii) the K-nearest-neighbour classifier and (iv) the feed-forward neural network classifier.

5.7.2 Description of the experiment

With this classification method, the defect classification stage is made up as a single multiclass classifier that assigns a defect type label among four to each feature vector fed into it.

The steps of the experiments are as follows:

1. Extract GLCM features for all the samples in the dataset.
2. Normalise the features of the training samples for each fabric type and for all the defect types together. This normalisation step should also provide the normalisation parameters to be used for the samples of the corresponding testing set.
3. For each fabric type, train a defect detector (binary classifier) using the normalised features of all the defect-free samples and the normalised features of all the defective samples, as well as their true class labels. This is the training phase of the defect detection stage.

4. Classify the samples of training set as defective or defect-free by feeding their normalised features into the detector trained in step 3.
5. Normalise the features of the testing samples using the normalisation parameters obtained from step 2.
6. Classify the samples of the testing set as defective or defect-free by feeding their normalised features into the defect detector trained in step 3.
7. Train the defect classifier by using the normalised features of the defective samples of the training set as well as their true class labels. This classifier is trained to be able to discriminate the four types of defects.
8. Classify each sample of the training set detected as defective in step 4 into one of the four types of defects.
9. Classify each sample of the testing set detected as defective in step 6 into one of the four types of defects.
10. Calculate the classification rate by dividing the number of samples of that fabric type correctly classified by the total number of samples of that fabric type in the dataset.
11. Compile the confusion matrix for that defect classification experiment.

The block diagram in Figure 5-18 illustrates how that classification is done once the two classifiers in the system are trained.

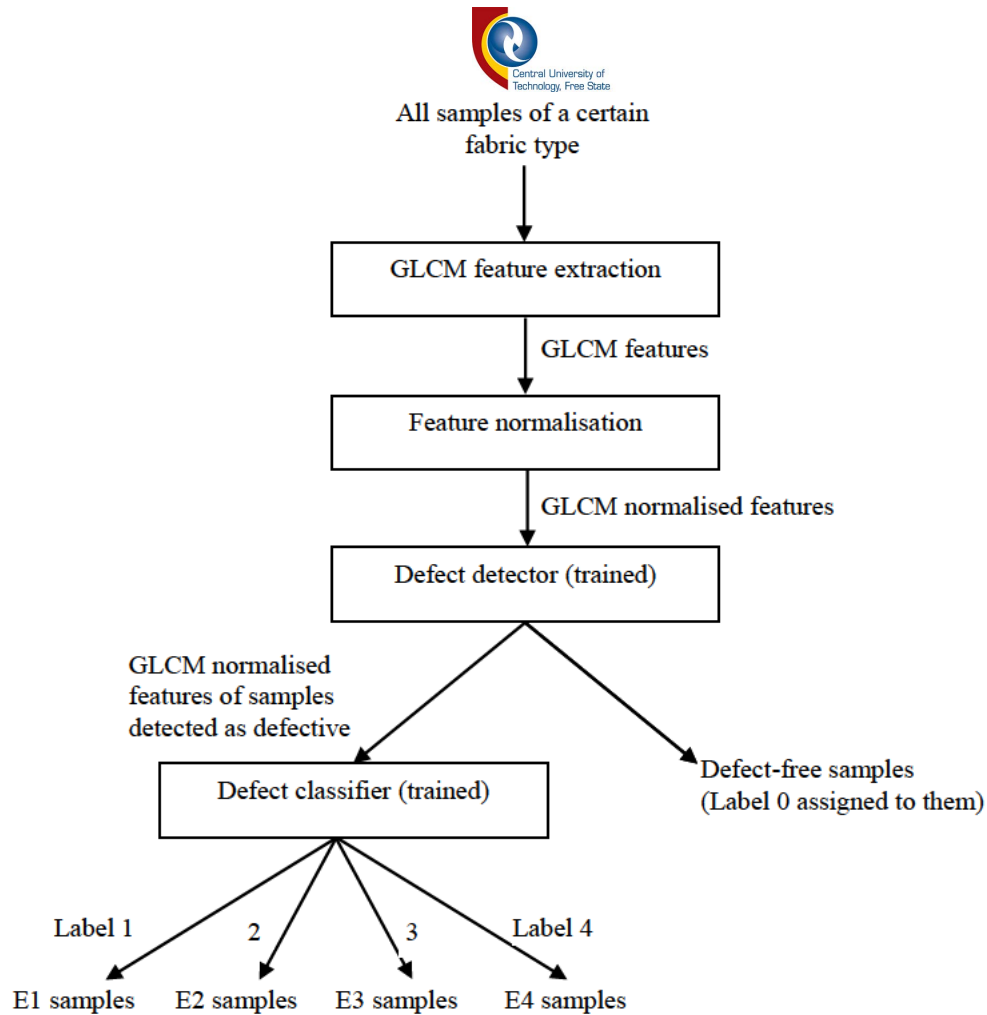


Figure 5-18: Single-level defect classification using GLCM features

5.7.3 Results and interpretation

Table 5-16 and Figure 5-19 shows the average classification rate of samples of each of the fabric types according to the defect types they contain among the five categories (i.e. E0, E1, E2, E3 and E4). One should remember that the defect type E0 means absence of any defect.

Table 5-16: Average defect classification rates (%) obtained using GLCM features

Fabric type	Classifier							
	ML		MCE		KNN		Neural	
	All	Opt	All	Opt	All	Opt	All	Opt
C1R1	70.8	66.9	75.9	73.5	80.9	73.4	82.0	81.6
C1R3	64.6	63.1	70.0	70.3	73.7	71.1	76.4	75.7
C2R2	56.9	51.5	61.2	61.0	66.1	62.8	66.9	65.2
C2R3	59.6	57.7	61.0	61.0	65.0	65.0	66.7	66.1
C3R1	47.6	47.0	50.0	47.2	65.0	63.0	71.3	68.8
C3R3	49.6	36.5	57.8	46.5	73.8	69.7	75.7	74.7
C4R1	36.6	35.6	43.3	41.1	63.1	58.4	67.4	65.3
C4R3	37.0	39.9	42.2	41.4	62.2	58.6	64.8	60.9

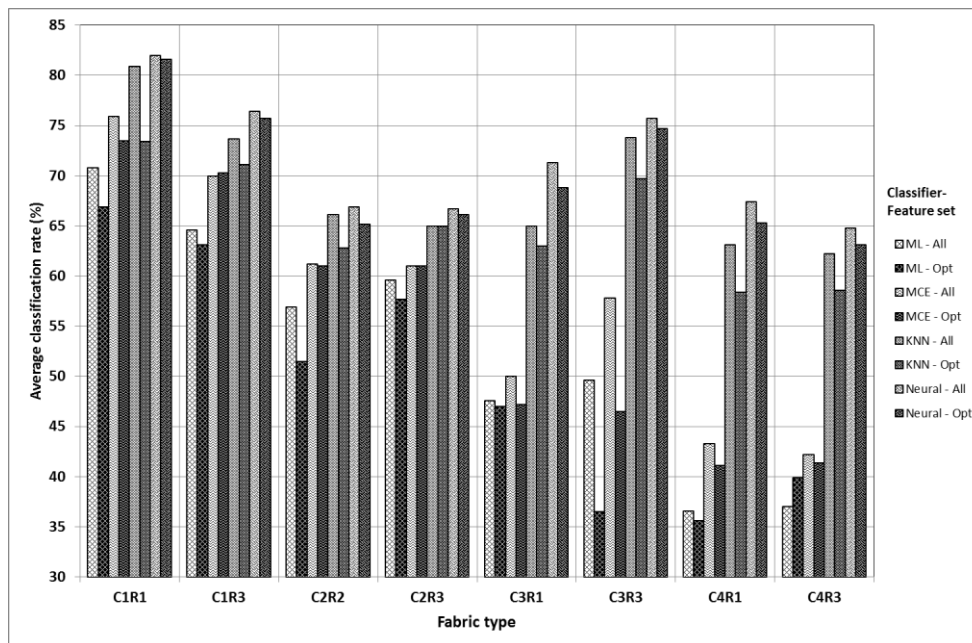


Figure 5-19: Average classification rate (%) for different types of fabrics using different classifiers. Comparison between the classification performance of the full set of fourteen features and the optimal feature set identified for defect detection

We can see from the results that the defect classification rates are lower than the defect detection rates found in previous sections of this chapter. This is normal, because the defect classifier has to discriminate among five classes, while the defect detector has to distinguish only two classes.

Comparing classifiers we see from the results that the feed-forward neural network is the best classifier, followed by the K-nearest-neighbour classifier (K=3 for the detector and K=9 for

the classifier), the MCE-trained Euclidean distance classifier and the ML-trained distance classifier. This is in agreement with the findings about defect detection in Subsection 5.6.2.

Comparing the defect classification rates of the optimal feature sets identified earlier for defect detection to those of the full set of fourteen features, the results show that the optimal features sets perform poorer than the full feature set. The only exceptions are for the fabric types C1R3 and C4R3, when the ML-trained Euclidean distance classifier is used. This means that those features are not optimal for defect classification. Therefore, the optimal features identified for defect detection should not be taken to necessarily also be optimal for defect classification. A new study should be undertaken to identify the optimal GLCM features for defect classification.

Table 5-17 shows four sample confusion matrices corresponding to the best classification rate and the worst classification rate of the results in Table 5-16 and Figure 5-19. From the confusion matrices on the right side of the table we can see that the low classification rates are mainly due to the fact that the ML-trained Euclidean distance classifier has great difficulties in distinguishing defect-free samples of fabric type C4R1 from samples with defects of type E1. This is shown by the off-diagonal elements (E0, E1) with high values of 1557 and 1388. The samples with the other defects are also generally confused because the off-diagonal elements are of the same size and even higher than the corresponding diagonal elements.

Looking at the confusion matrices on the left side of the table, we see that the off-diagonal are relatively low compared to the corresponding diagonal elements. This shows the ability of the classifier in discriminating the different defect types. Improvement efforts would focus first on differentiating better E2 from E4, as the confusion matrices show that many samples are classified as E4 when in fact they are E2.

Table 5-17: Sample confusion matrices for defect classification using GLCM features

	Highest rate C1R1 using a feed-forward neural network classifier						Lowest rate C4R1 using a ML-trained Euclidean distance classifier							
Optimal feature set	when their true class is						when their true class is							
		E0	E1	E2	E3	E4		E0	E1	E2	E3	E4		
	Classified as	E0	4747	3	90	53	98	Classified as	E0	2267	1557	394	79	447
		E1	24	239	24	56	75		E1	217	495	64	29	22
		E2	167	4	801	14	98		E2	749	319	354	19	212
		E3	88	36	77	164	86		E3	386	335	55	86	35
		E4	227	22	189	40	574		E4	514	499	149	30	175
	Classification rate: 81.6%						Classification rate: 35.6%							
All fourteen GLCM features	when their true class is						when their true class is							
		E0	E1	E2	E3	E4		E0	E1	E2	E3	E4		
	Classified as	E0	4719	9	118	26	119	Classified as	E0	2123	1388	336	98	799
		E1	34	299	12	32	41		E1	191	484	51	34	67
		E2	146	7	860	7	64		E2	492	262	331	2	566
		E3	93	82	86	113	77		E3	390	326	30	101	50
		E4	224	43	191	25	569		E4	350	411	149	24	433
	Classification rate: 82.0%						Classification rate: 36.6%							

5.8 Modified GLCHS for fast computation of GLCM features

The grey level co-occurrence matrix is generally a sparse matrix (with only a few nonzero elements). For example, a 32x32 pixels 64-grey level image patch has $64^2 = 4096$ elements, yet the number of nonzero elements cannot exceed $31 \times 32 = 992$. On the other hand, as shown by equations (3.20) through (3.42), the computation of any of the GLCM features involves the multiplication by the grey level co-occurrence probability and the summation over all elements of the GLCM. Therefore there would be a great computational gain if the information in the GLCM was represented in such a way that the calculations of the GLCM features are done only over nonzero elements.

In this regard, Clausi and Jernigan [118] proposed a method they called grey level co-occurrence linked list (GLCLL), where only nonzero grey level probabilities were stored in a sorted linked list. Their method was improved by Svolos and Todd-Pokropek [119], who represented the same information in a tree data structure. Since GLCLL requires maintaining a sorted list, Clausi and Zhao [120] dropped the use of the sorted linked list and instead used the combination of a hash table and a linked data structure. They called the new improved method grey level co-occurrence hybrid structure (GLCHS).

GLCHS is based on the combination of a two-dimensional hash table and a linked list. The hash table has the same size as the GLCM and contains in each cell a pointer to a node in the linked list and the co-occurrence probability corresponding to that cell. Only cells with nonzero co-occurrence probabilities have corresponding nodes in the linked list, all pointers in all the other cells are set to “NULL”. The pointed node in the linked list contains the grey level pair and two pointers, one to the previous list in the linked list and the other to the next node. In addition, two pointers to point the head and the tail of the linked list are provided. Figure 5-20 illustrates this situation. The hash table is used for easy access to the nodes of the linked list during the compilation of the co-occurrences probabilities, while the linked list is used during the calculation of the co-occurrence features moving rapidly from the head to the tail of the list. The list is double linked to allow easy insertion of new nodes during the compilation of the co-occurrence probabilities.

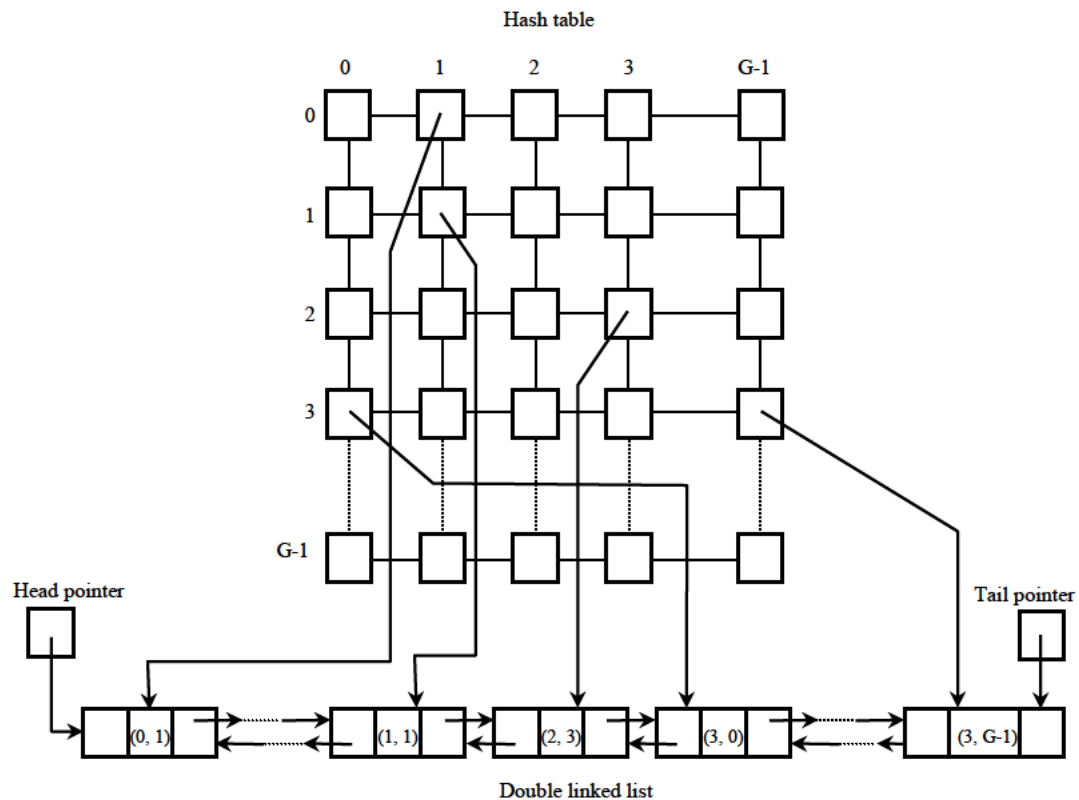


Figure 5-20: Data structures of GLCHS

We modify GLCHS for fast implementation using MATLAB as follows.

1. The hash table is replaced with a matrix “POINTERS” of the same size whose entries are the row indexes of another matrix “DATA” whose entries are the actual co-occurrence data.

2. The matrix “DATA” has three columns, the first two columns contain the grey level pair (I, J), while the third column contains the number of times the grey pair appears in the image. Grey pairs that do not appear in the image are not represented in that matrix.
3. At the beginning of the compilation of the co-occurrences probabilities, all the cells of the matrix “POINTERS” are initialised to zero. The matrix “DATA” is also created with the maximum possible number of rows (GxG), where G is the number of grey levels used to represent the image. All the cells of matrix “DATA” are also initialised to zero. A variable, ‘CurrentRowPointer’, to contain the row number where new data is to be written is created and initialised to value 1.
4. The algorithm continues as follows:
 - a. The image (or image patch) is scanned from the top left corner to the bottom right corner, one line at a time from left to right. In this process, each pixel is considered along with the corresponding pixel that together makes up a grey level co-occurrence pair. For example, if the interpixel distance $d=1$ and the interpixel orientation $\theta=0^\circ$, then pixel (i, j) is considered along with pixel (i, j+1). The grey level values of the two pixels are read and used to update the matrices “POINTERS” and “DATA”.
 - b. If it is the first time that the specific pair of grey level values is encountered, then the corresponding cell in the matrix “POINTERS” still has the value 0. If so, the grey level values of the pair are written in the first and second column of the current row of the matrix “DATA” (indicated by the variable ‘CurrentRowPointer’), while the third column is set to 1 to indicate that so far there is one occurrence of that grey level pair. Thereafter, the relevant cell in the “POINTERS” matrix is set to the current value of the variable ‘CurrentRowPointer’ and then that variable is incremented to point the next available row in the matrix “DATA”.
 - c. If it is not the first time that the pair of grey level value is encountered, then the corresponding cell in the matrix “POINTERS” contains the row number of the corresponding entry in the matrix “DATA”. That information is used to access the row and increment the value of its third column.
 - d. When all the pixels of the image have been considered, the unused rows of the matrix “DATA” are deleted. Thereafter, the sum of values in the third

column of that matrix is calculated. That sum is the total number of pixel pairs in the image. Each of the values of the third column of the matrix “DATA” are divided by their sum, and the result, which represents the relative occurrence (or co-occurrence probability) of the corresponding grey level pair, is stored in the same column replacing the absolute number of occurrences of the pair.

5. The output of that algorithm is the matrix “DATA” with three columns. The first two columns contain the values of the grey level pairs while the third column contains their relative occurrence (co-occurrence probabilities) in the image. That information is then used to compute the co-occurrence features according to equations (3.20) through (3.37).

The main difference between our algorithm and the original GLCHS is that the data structures are implemented using matrices rather than linked lists. This allows easier and faster implementation in MATLAB.

The algorithm was used to calculate the fourteen GLCM features of 32x32 fabric image patches with 64 grey levels, and the average execution time was 42.3% of the corresponding implementation that uses the grey level co-occurrence matrix.

5.9 Summary

In this chapter we evaluated the performance of the texture features derived from the grey level co-occurrence matrix for fabric defect detection and classification. The objective was to identify, among the fourteen features proposed by Haralick et al. [31], the most suitable subset for defect detection for different types of fabrics. We also dealt with the problem of choosing appropriate parameters (quantisation level and intersample distance) for efficient extraction of GLCM features.

For the problem of identifying the best feature set for each fabric type, we used the following procedure. After realising that the total number of possible combinations of GLCM features was not too high, we decided to perform an exhaustive search. Therefore, experiments to evaluate the defect detection rate using each of the fourteen features individually were performed and the feature that yielded the best detection rate for each fabric type and each type of defect was noted. We then performed similar experiments using all 91 possible pairs of features, recording the pair with the best rate for each fabric type and each type of defect, then using all the 364 possible triplets and noting the triplet that yielded the best rate. We continued this procedure until all the possible combinations of features were tried. After that,

we plotted a graph showing the variation of the average best detection rate (across different types of fabrics) with respect to the number of features constituting the experimental feature set. This allowed us to realise that the number of features that yields the best detection rate is **three** when a ML-trained Euclidean distance classifier was used.

After determining the optimal number of features in a feature set, we devised a procedure to identify the actual features for each type of fabric. The optimal feature sets were then identified and recorded for each of the fabric types in the dataset.

The above observations were made using a ML-trained Euclidean distance classifier. In order to verify their validity with different classifiers, we performed the same experiments with (i) a MCE-trained Euclidean distance classifier, (ii) a K-nearest-neighbour classifier (K=3), and (iii) a feed-forward neural network classifier. We realised that the optimal number of features was classifier dependent. That number was found to be **five** for both the MCE-trained Euclidean distance classifier and the K-nearest-neighbour classifier (K=3) and **nine** for the feed-forward neural network. These results seem to indicate that the more accurate a classifier, the more features it is capable of handling effectively.

To identify the best quantisation level of the original image, we performed the defect detection experiments with original fabric images quantised using 16, 32, 64, 128 and 256 grey levels and recorded the resulting defect detection rates. After analysing the results we observed that the optimal choice of quantisation is achieved with 64 grey levels. We also performed experiments to identify the best intersample distance for the compilation of the grey level co-occurrence matrices. Among ten distances from 1 through 10 pixels, the number 1 emerged as the best choice for most of the fabric types.

The classification related experiments revealed that the optimal features identified for the defect detection experiments were not necessarily the same for discriminating among the different types of defects. In fact, the optimal features identified during the defect detection experiments performed consistently poorer than the full set of features and for all the classifiers we used. More studies are needed to identify the best features for the classification experiments.

Finally, we proposed the modified GLCHS algorithm for fast computation of GLCM features for implementation in MATLAB.

Chapter 6: Exploration of wavelet transform features

6.1 Introduction

The wavelet transform is viewed as the most promising tool for fabric defect detection and classification [16]. In this thesis we will combine texture features extracted using the wavelet transform with features extracted from the grey level co-occurrence matrix and Markov random field model-based features for the purpose of defect detection and classification. The objective is to find a method that is more effective in terms of detection and classification rates than any of those methods taken separately. In this chapter, we study the wavelet features so that we can select from them a minimum subset that can be quickly computed without sacrificing their texture discriminating ability.

This chapter is organised as follows. In Section 6.2, we choose the type of wavelet transform to use. We choose the DTCWT and UDWT due to their shift-invariance property. Section 6.3 deals with the design of the wavelets to implement UDWT, while Section 6.4 deals with the design of the complex wavelet for the DTCWT implementation. In Section 6.4 we investigate the wavelet features to be used, while in Section 6.6 we describe the algorithm for their extraction. Section 6.7 compares the defect detection performance of UDWT- and DTCWT-based features on average and for each fabric type of our dataset. In sections 6.8 and 6.9, we study the variation of the defect detection rates achieved by wavelet features with respect to their wavelet size of support and with respect to the number of decomposition levels respectively. Section 6.10 summarises the findings of sections 6.7 through 6.9, identifying the best wavelet features for each type of fabrics. Experiments for sections 6.7 through 6.9 are done using the MCE-trained Euclidean distance classifier. In Section 6.11 we extend those experiments to the K-nearest-neighbour classifier and the feed-forward neural network classifier. Section 6.12 deals with the problem of defect classification using the wavelet features and Section 6.13 summarises the chapter.

6.2 Which wavelet transform to use?

As described in Section 3.3, several types of wavelet transform exist and a few of them have been used in research related to fabric defect detection and classification.

The continuous wavelet transform (CWT) leads to a continuous representation of a signal in the scale-time domain or of an image in the scale-space domain. Although it allows a fine exploration of the signal (or image) behaviour through a scale range, it is a highly redundant representation. Thus it is not suitable for images, as the computation cost of decomposition and reconstruction would be too high [174].

For applications involving wavelet transform of images it is preferable to use the discrete wavelet transform (DWT) rather than CWT, and generally its simple dyadic version presented in (3.46) through (3.48) extended to two dimensions. However, the DWT in its dyadic version is not shift invariant and is not therefore appropriate for pattern recognition applications such as fabric defect detection and classification. For such application the undecimated version (UDWT) is more suitable.

Even if the UDWT solves the problem of shift variance, it has a high redundancy rate of $3L+1$ for image representation, where L is the number of wavelet decomposition levels. That leads to increased computational requirements [99]. In addition, similar to DWT, UDWT has the shortcoming of poor directional selectivity for diagonal features in 2D. This weakness of UDWT lowers the discrimination power of its derived texture features. The dual-tree complex wavelet transform (DTCWT), first introduced by Kingsbury in 1998 [99-100], is approximately shift invariant and makes it possible to get directional wavelets in two and more dimensions with only $2x$ redundancy in 1-D (2^d for d -dimensional signals, in general) [72]. In addition, the DTCWT allows extracting features that are approximately rotationally invariant [72]. This is important when the orientation of images under study is not known a priori. The fabric images in our dataset are part of this case as they were arbitrarily rotated.

In this chapter we will study the performance of the features from both the UDWT and DTCWT, making a comparison between the two types. We will study their performance in terms of fabric defect detection and classification rates considering three aspects: (i) the specific selected features, (ii) the size of wavelet support, and (iii) the number of wavelet decomposition levels as well as their combination.

6.3 Design of the wavelet for the UDWT

The UDWT is usually implemented using a two-channel nonsampled filter bank illustrated in Figure 6-1. $H_0(z)$ and $F_0(z)$ are the transfer functions of the low-pass decomposition and reconstruction filters respectively, while $H_1(z)$ and $F_1(z)$ are the transfer functions of the high-pass decomposition and reconstruction filters respectively. $X(z)$ is the z -transform of the input signal while $Y(z)$ is the z -transform of the reconstructed signal.

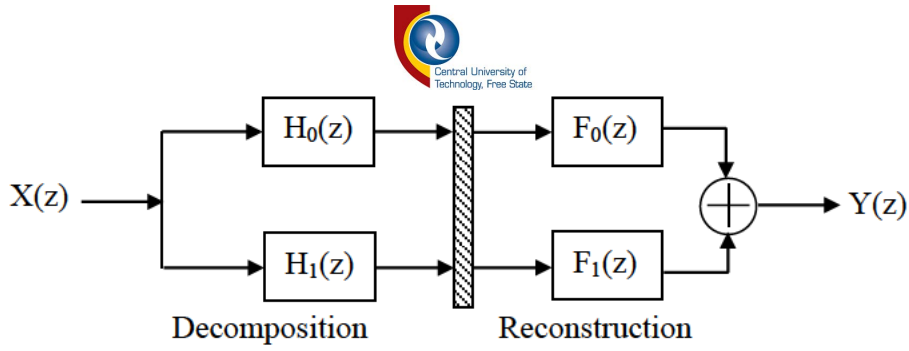


Figure 6-1: A two-channel nonsampled filter bank

Perfect reconstruction is possible if (6.1) is satisfied.

$$H_0(z)F_0(z) + H_1(z)F_1(z) = 1 \quad (6.1)$$

Cvetković and Vetterli [175-176] showed that the analysis and synthesis filters can be finite impulse response (FIR) if and only if the analysis filters are power complementary as described by (6.2).

$$H_0(z)H_0(z^{-1}) + H_1(z)H_1(z^{-1}) = 1 \quad (6.2)$$

Then the frame implemented by the filter bank is tight and the reconstruction filters are the time reversed versions of the decomposition filters as given by (6.3) and (6.4).

$$F_0(z) = H_0(z^{-1}) \quad (6.3)$$

$$F_1(z) = H_1(z^{-1}) \quad (6.4)$$

The regularity of derived wavelet is related to the multiplicity of zeros of the low-pass filter $H_0(z)$ at $z=-1$. $H_0(z)$ should have at least one zero at $z=-1$, and highly regular wavelets are obtained from filters with a high number of zeros at $z=-1$ [175]. The number of vanishing moments of the wavelet is determined by the number of zeros of the high-pass filter $H_1(z)$ at $z=-1$.

Our design method assumes that regular wavelets with a high number of vanishing moments lead to better results. Therefore, for a given filter length we will maximise the number of zeros of $H_0(z)$ and the number of zeros of $H_1(z)$ at $z=-1$ using the method provided by Cvetković and Vetterli [176]. Such filters are called maximally flat filters.

Assuming that we wish to have filters of length L , with N_0 zeros of $H_0(z)$ at $z=-1$ and N_1 zeros of $H_1(z)$ at $z=-1$, Cvetković and Vetterli [176] showed that it is possible to design a filter pair $H_0(z)$ and $H_1(z)$ of length L satisfying (6.2) such that $1 \leq N_0 < L$, $1 \leq N_1 < L$, $N_0 + N_1 \leq L$.

As we wish to maximise N_0+N_1 for a given filter length L , then $N_0+N_1=L$. In this case the low-pass filter of the maximally flat pair $H_0(z)$ can be found by spectral factorisation of the product $H_0(z)H_0(z^{-1})$ that is obtained using (6.5)

$$H_0(z)H_0(z^{-1}) = P\left(\left(\frac{1-z}{2}\right)\left(\frac{1-z^{-1}}{2}\right)\right) \quad (6.5)$$

where $P(y)$ is the polynomial obtained using (6.6)

$$P(y) = (1-y)^{N_0} \left(\sum_{l=0}^{L-1-N_0} \binom{N_0+l-1}{l} y^l \right) \quad (6.6)$$

The roots of $H_0(z)H_0(z^{-1})$ are necessarily pairs of reciprocals with one member of the pair inside or on the unit circle in the z plane, while the other is outside or on the unit circle. Spectral factorisation means that the polynomial is factorised into two factors so that the roots of one factor are the ones inside or on the unit circle, while the roots of the other factor are the ones outside or on the unit circle.

The high-pass filter $H_1(z)$ is obtained by spectral factorisation of the polynomial obtained using (6.7).

$$H_1(z)H_1(z^{-1}) = 1 - P\left(\left(\frac{1-z}{2}\right)\left(\frac{1-z^{-1}}{2}\right)\right) \quad (6.7)$$

Using the method described above we obtained the filter coefficients shown in Table 6-1 for an even length filter of length varying from 2 to 16. For each filter length L , N_0 is chosen equal to N_1 if L is even. We calculated even length filters only so that we can compare their performance with the dual-tree wavelet filters which are designed for even length only as shown in the next section.

Table 6-1: Decomposition filters h0 and h1 for UDWT wavelets

Wavelet 0		Wavelet 1		Wavelet 2		Wavelet 3	
h0	h1	h0	h1	h0	h1	h0	h1
0.5	0.5	0.341506	0.341506	0.235234	0.235234	0.162902	0.162902
0.5	-0.5	0.591506	-0.59151	0.570558	-0.57056	0.505473	-0.50547
Wavelet 7		0.158494	0.158494	0.325183	0.325183	0.4461	0.4461
h0	h1	-0.09151	0.091506	-0.09547	0.095467	-0.01979	0.019788
0.038478	0.038478	Wavelet 6		-0.06042	-0.06042	-0.13225	-0.13225
0.221234	-0.22123	h0	h1	0.024909	-0.02491	0.021808	-0.02181
0.477743	0.477743	0.05505	0.05505	Wavelet 5		0.023252	0.023252
0.413908	-0.41391	0.280396	-0.2804	h0	h1	-0.00749	7.49E-03
-0.01119	-0.01119	0.515574	0.515574	0.078871	0.078871	Wavelet 4	
-0.20083	0.200829	0.332186	-0.33219	0.349752	-0.34975	h0	h1
0.000334	0.000334	-0.10176	-0.10176	0.531132	0.531132	0.113209	0.113209
0.091038	-9.10E-02	-0.15842	0.158418	0.222916	-0.22292	0.426972	-0.42697
-1.23E-02	-1.23E-02	0.050423	0.050423	-0.15999	-0.15999	0.512163	0.512163
-3.12E-02	3.12E-02	5.70E-02	-5.70E-02	-0.09176	0.091759	0.097883	-0.09788
9.89E-03	9.89E-03	-0.02689	-2.69E-02	0.068944	6.89E-02	-0.17133	-0.17133
6.18E-03	-6.18E-03	-1.17E-02	1.17E-02	0.019462	-1.95E-02	-0.0228	0.022801
-3.44E-03	-3.44E-03	8.87E-03	8.87E-03	-2.23E-02	-2.23E-02	0.054851	5.49E-02
-2.77E-04	2.77E-04	3.04E-04	-3.04E-04	0.000392	-3.92E-04	-0.00441	4.41E-03
4.78E-04	4.78E-04	-1.27E-03	-1.27E-03	3.38E-03	3.38E-03	-0.0089	-8.90E-03
-8.31E-05	8.31E-05	2.50E-04	-2.50E-04	-7.62E-04	7.62E-04	2.36E-03	-2.36E-03

6.4 Design of the dual-tree complex wavelet filters

6.4.1 Fundamentals of dual-tree complex wavelet design

The dual-tree complex wavelet transform allows obtaining complex wavelet coefficients that are approximately shift-invariant. This is accomplished by using two real wavelets, $\psi_h(t)$ and $\psi_g(t)$. The two wavelets are designed so that $\psi_g(t)$ is approximately the Hilbert transform of $\psi_h(t)$. The resulting complex wavelet $\psi_c(t)$, described by (6.8) is therefore approximately analytic [72]. That means that its Fourier transform is approximately 0 for one half of the frequency axis.

$$\psi_c(t) = \psi_h(t) + j\psi_g(t) \quad (6.8)$$

Each of the two real wavelets $\psi_h(t)$ and $\psi_g(t)$ is implemented using a two-channel filter bank so that a multilevel wavelet decomposition is implemented by a dual-tree structure as shown in Figure 6-2.

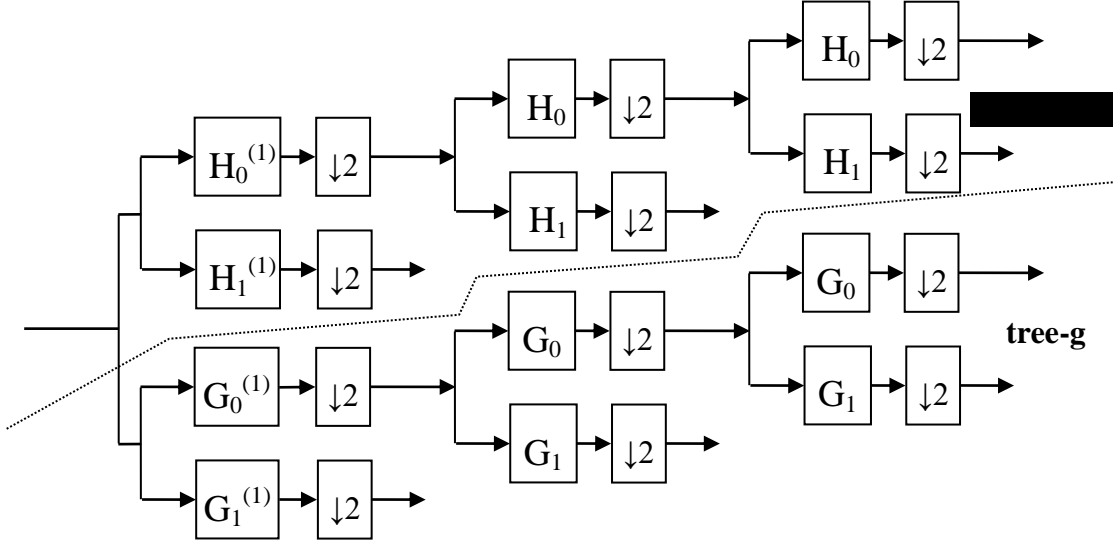


Figure 6-2: Implementation of 1D dual-tree complex wavelet transform using FIR real coefficient filters

H_0 and H_1 are respectively the low- and high-pass decomposition filters for tree-h (real part of the complex wavelet) while G_0 and G_1 are respectively the low- and high-pass decomposition filters for tree-g (imaginary part of the complex wavelet). The first level filters ($H_0^{(1)}$, $H_1^{(1)}$) and ($G_0^{(1)}$, $G_1^{(1)}$) can be any set of orthogonal or biorthogonal wavelet filters [72].

The design of the filters should be done to satisfy the perfect reconstruction condition for both tree-h and tree-g, as well as the approximate Hilbert transform status of tree-g with respect to tree-h. Assuming that the first condition of perfect reconstruction is satisfied, it has been shown that to satisfy the second condition (Hilbert transform), one of the low-pass filters H_0 and G_0 should be approximately a half-sample shift of the other as described by (6.9) .

$$g_0[n] \approx h_0[n - 0.5] \quad (6.9)$$

The relation (6.9) in the time domain is equivalent to (6.10) in the frequency domain.

$$G_0(e^{j\omega}) = e^{-j\frac{\omega}{2}} H_0(e^{j\omega}) \quad (6.10)$$

When written in terms of magnitude and phase (6.10) is equivalent to (6.11) and (6.12).

$$|G_0(e^{j\omega})| = |H_0(e^{j\omega})| \quad (6.11)$$

$$\angle G_0(e^{j\omega}) = \angle H_0(e^{j\omega}) - 0.5\omega \quad (6.12)$$

Therefore designing for the DTCWT is equivalent to designing two two-channel filter banks whose filters possess the following desired properties [72]:

1. Approximate half-delay property
2. Perfect reconstruction (orthogonal or biorthogonal)
3. Finite support (FIR filters)
4. Vanishing moments (good stopband)

The best known among the proposed solutions are (i) linear phase biorthogonal solution [99-100, 177-178], (ii) q-shift solution [179-180] and (iii) common-factor solution [181]. We will adopt the q-shift solution in this thesis because it allows a more symmetric subsampling structure [179] and is simpler to design – only one filter needs to be designed. In addition, the obtained filters are orthogonal, which makes the transform a tight frame which conserves energy from the signal to the transform domain [180].

6.4.2 Q-shift design solution

With q-shift design the low-pass filters H_0 and G_0 are chosen to have even length and to be time reverse of each other as described by (6.13).

$$g_0[n] = h_0[N - 1 - n], \text{ with } N \text{ even} \quad (6.13)$$

Under such conditions the magnitude condition (6.11) is exactly satisfied but the phase condition (6.12) is not. The q-shift solution approximates the phase condition by a single filter H_0 , designed to have a group delay of approximately $\frac{1}{4}$ sample and to satisfy the perfect reconstruction conditions. As filter G_0 is the time reverse of H_0 , its group delay is approximately $\frac{3}{4}$ sample. This leads to the required difference of $\frac{1}{2}$ sample. Kingsbury [180] proposed a design method that allows obtaining linear phase complex wavelets and scaling function, an important property for image processing applications. We will adopt this method in this thesis.

To obtain the second property of perfect reconstruction with orthogonality and no aliasing for the filter bank, the standard conditions given by (6.14) through (6.17) are used.

$$G_1(z) = z H_0(-z) \quad (6.14)$$

$$H_1(z) = z^{-1} G_0(-z) \quad (6.15)$$

$$H_0(z)G_0(z) + H_0(-z)G_0(-z) = 2 \quad (6.16)$$

$$G_0(z) = H_0(z^{-1}) \quad (6.17)$$

To obtain an approximate delay of $\frac{1}{4}$ sample for $H_0(z)$, a length $4N$ linear phase (symmetric) low-pass filter $H_{L2}(z)$ is designed to operate at twice the required rate and then subsampled by 2:1. As the filter $H_{L2}(z)$ has a delay of $\frac{1}{2}$ its sample rate, the filter $H_0(z)$ will have half of its delay ($\frac{1}{4}$ sample) and twice its bandwidth as long as the filter $H_{L2}(z)$ has negligible gain between $\frac{1}{4}$ and $\frac{1}{2}$ of its sample rate. The details of the design can be found in [180].

Using this design method as well as the MATLAB program provided by Kingsbury [182], filters of length 4 through 14 were designed. They are shown in Table 6-2. Only filters $h_0[k]$ as shown because filters $g_0[k]$ are simply the time-reversed versions of corresponding filters $h_0[k]$. Once the filters are calculated, the extension to two dimensions is done as described in Subsection 3.3.8.

Table 6-2: Filters h_0 for different complex wavelets

<u>h0 for Wavelet 1</u>	<u>h0 for Wavelet 2</u>	<u>h0 for Wavelet 3</u>
0.163779031979466	-0.0857079630841502	-0.0690438428366266
0.816139266740385	0.1913007969323510	-0.0506761525555254
0.543327749207082	0.8218333624947300	0.2813666512150030
-0.109032485553837	0.5288071149925820	0.7565668459089610
<u>h0 for Wavelet 6</u>	-0.0290186182240319	0.5797556204871810
0.0017398739681362	-0.0130011307383845	-0.0721944491507631
-0.0027459325505043	<u>h0 for Wavelet 5</u>	-0.0849716476790101
0.0130575675739475	0.0003832197815069	0.0734105369838758
-0.0236356620546165	0.0264837358621132	<u>h0 for Wavelet 4</u>
-0.1061549045771790	-0.0050247175139210	0.0406676426878204
0.2450419775540350	-0.1004719196058730	-0.0100983326757664
0.7811979647922610	0.2380935587694480	-0.1065465695674370
0.5571491131791900	0.7869918742072030	0.2525439510986830
0.0002074202558124	0.5539569455120210	0.7752771348740080
-0.0814892321299389	-0.0068614508914790	0.5600544436286330
0.0198978800325300	-0.0788381045593577	-0.0013357598095111
0.0145845889663273	0.0009433604924234	-0.0915446472694999
-0.0028390208589604	-0.0014641208031492	-0.0009556669983319
-0.0017980717779448	0.0000211811221590	-0.0038486335955013

6.5 Which wavelet features to use?

As discussed in Subsection 3.3.9, the most commonly used wavelet features are (i) the channel variances or mean energies of coefficients of a sub-image, (ii) the mean absolute values of wavelet coefficients, (iii) entropies of magnitudes of wavelet coefficients, (iv) mean magnitude of complex wavelet coefficients and (v) variance of magnitude of wavelet coefficients.

The mean energies and the mean absolute values carry fundamentally the same information, and therefore it would not be useful to combine them. In our case, the sample size is limited to 32x32 pixels and thus the number of wavelet coefficients in a channel is at most $16^2=256$. Due to that small number of coefficients, we have realised that the entropy of coefficients does not vary much from one channel to another. Such a feature would not be discriminating enough. Therefore we will compare the performance of the two following sets of features: (i) the mean energies of wavelet coefficients combined with the variances of energy of wavelet coefficients, and (ii) the mean absolute values of wavelet coefficients combined with the variances of absolute values of wavelet coefficients.

6.5.1 Feature sets for UDWT

Every sample will be submitted to five levels of UDWT decomposition. As there are three wavelet channels per level of decomposition (horizontal, vertical and diagonal), the number of features extracted from each channel is to be multiplied by 15 to get the total number of extracted features for a given sample. Therefore, these chosen wavelet features can be expressed by (6.18) through (6.21) for the UDWT.

In (6.18) through (6.21), $k=1, 2, 3$ denotes the wavelet channel, 1=horizontal and 2=vertical, 3=diagonal. $l = 1, 2, \dots, 5$ denotes the level of wavelet decomposition, while $N=32$ denotes the sample width in pixels. The wavelet coefficients are denoted by $dr_{i,j}^{l,k}$.

- Mean energy of wavelet coefficients:

$$f_{r1}^{l,k} = \frac{1}{N^2} \sum_{i=1}^N \sum_{j=1}^N (dr_{i,j}^{l,k})^2 \quad (6.18)$$

- Variance of energies of wavelet coefficients:

$$f_{r2}^{l,k} = \frac{1}{N^2} \sum_{i=1}^N \sum_{j=1}^N \left[(dr_{i,j}^{l,k})^2 - f_{r1}^{l,k} \right]^2 \quad (6.19)$$

- Mean absolute values of wavelet coefficients:

$$f_{r3}^{l,k} = \frac{1}{N^2} \sum_{i=1}^N \sum_{j=1}^N |dr_{i,j}^{l,k}| \quad (6.20)$$

- Variance of absolute values of wavelet coefficients:

$$f_{r4}^{l,k} = \frac{1}{N^2} \sum_{i=1}^N \sum_{j=1}^N \left(|dr_{i,j}^{l,k}| - f_{r3}^{l,k} \right)^2 \quad (6.21)$$

The two feature sets to be compared in our experiments are given by (6.22) and (6.23).

- (i) Set r1: the means and variances of energies of wavelet coefficients.

$$\text{Set } r1 = \{f_{r1}^{l,k}, f_{r2}^{l,k}\}, l = 1, 2, \dots, 5 \quad k = 1, 2, 3 \quad (6.22)$$

- (ii) Set r2: the means and variances of absolute values of wavelet coefficients.

$$\text{Set } r2 = \{f_{r3}^{l,k}, f_{r4}^{l,k}\}, l = 1, 2, \dots, 5 \quad k = 1, 2, 3 \quad (6.23)$$

6.5.2 Feature sets for DTCWT

Similar to UDWT, every sample will be submitted to five levels of DTCWT wavelet decomposition. However, each level of decomposition produces six wavelet channels corresponding to the six orientations -75° , -45° , -15° , $+15^\circ$, $+45^\circ$ and $+75^\circ$. Therefore, the number of features extracted per channel is to be multiplied by 30 to get the total number of features extracted per sample.

The features we will use for DTCWT can be expressed by (6.24) through (6.27). In these equations, $d=1, 2, \dots, 6$ denotes the wavelet channel (i.e. orientation of the wavelet), $1=-75^\circ$, $2=-45^\circ$, $3=-15^\circ$, $4=+15^\circ$, $5=+45^\circ$ and $6=+75^\circ$. $l = 1, 2, \dots, 5$ denotes the level of wavelet decomposition, while $N=32$ denotes the sample width in pixels. The complex wavelet coefficients are denoted by $dc_{i,j}^{l,k}$.

- Mean energy of complex wavelet coefficients:

$$f_{c1}^{l,k} = \frac{(l+1)^2}{N^2} \sum_{i=1}^{N/(l+1)} \sum_{j=1}^{N/(l+1)} |dc_{i,j}^{l,k}|^2 \quad (6.24)$$

- Variance of energies of complex wavelet coefficients:

$$f_{c2}^{l,k} = \frac{(l+1)^2}{N^2} \sum_{i=1}^{N/(l+1)} \sum_{j=1}^{N/(l+1)} \left(|dc_{i,j}^{l,k}|^2 - f_{c1}^{l,k} \right)^2 \quad (6.25)$$

- Mean magnitude of complex wavelet coefficients:

$$f_{c3}^{l,k} = \frac{(l+1)^2}{N^2} \sum_{i=1}^{N/(l+1)} \sum_{j=1}^{N/(l+1)} |dc_{i,j}^{l,k}| \quad (6.26)$$

- Variance of magnitudes of complex wavelet coefficients:

$$f_{c4}^{l,k} = \frac{(l+1)^2}{N^2} \sum_{i=1}^{N/(l+1)} \sum_{j=1}^{N/(l+1)} \left(|dc_{i,j}^{l,k}| - f_{c3}^{l,k} \right)^2 \quad (6.27)$$

The two feature sets to be compared in our experiments are given by (6.28) and (6.29).

- (iii) Set c1: the means and variances of energies of complex wavelet coefficients.

$$\text{Set } c1 = \{f_{c1}^{l,k}, f_{c2}^{l,k}\}, l = 1, 2, \dots, 5 \quad d = 1, 2, \dots, 6 \quad (6.28)$$

- (iv) Set c2: the means and variances magnitudes of complex wavelet coefficients.

$$\text{Set } c2 = \{f_{c3}^{l,k}, f_{c4}^{l,k}\}, l = 1, 2, \dots, 5 \quad d = 1, 2, \dots, 6 \quad (6.29)$$

6.5.3 Feature normalisation

After extraction, the wavelet features will be normalised as described in Subsection 3.5.2 before being fed into a classifier. The type of normalisation to use will depend on the kind of classifier used.

For the Euclidean distance classifiers, whether trained using the ML or MCE algorithms, and for K-nearest-neighbour classifier, we will use the min-max normalisation as given by (3.118). For the feed-forward neural network classifier, we will use the min-max normalisation that linearly scales the input data into the range [-1, 1] rather than [0, 1] using the MATLAB function '*mapminmax*'. This is because the sigmoid transfer functions used in our neural networks have input ranges that are symmetric with respect to 0.

6.6 Feature extraction algorithm

Defect detection is done by extracting features from a sample and then feeding the extracted features into a classifier that decides whether the sample is defective or defect-free. In this subsection we deal with the feature extraction part, while the decision part will be dealt with subsequently.

Every fabric image has 384 elementary samples, but only a few of them are used in the experiments as previously explained in sections 4.3 through 4.5. However, for wavelet-based feature extraction the wavelet transform is done for the whole image and then only the wavelet coefficients corresponding to the elementary samples of interest are taken from the resulting subbands for feature calculation. That was done to avoid border problems that would have resulted from performing the wavelet transform from each elementary sample.

6.6.1 Feature extraction algorithm for UDWT

The algorithm we used for feature extraction from a sample using the undecimated wavelet transform is shown in Table 6-3 as pseudo code. That algorithm was performed for each of the eight wavelets designed in Section 6.3 because we used the results to compare the performance of the different wavelets.

The UDWT wavelet decomposition in step 5) was implemented using the MATLAB function '*ndwt2*' which performs the 2D non-decimated wavelet decomposition of an image given the desired number of decomposition and the low-pass and high-pass decomposition filters *h0* and *h1* respectively.

Table 6-3: Algorithm for feature extraction using UDWT

1)	For each of the eight fabric types (C1R1, C1R3, C2R2, C2R3, C3R1, C3R3, C4R1 and C4R3)
2)	For each of the four defect types (E1, E2, E3 and E4)
3)	For each of the two sample sets (training and testing)
4)	For each of the 50 images
5)	Perform the five level UDWT decomposition of the image, obtaining the horizontal, vertical and diagonal subbands for each of the five levels as a result.
6)	For each of the five levels
7)	For each subband (horizontal, vertical and diagonal)
8)	Divide the subband into 32x32 non-overlapping windows.
9)	Calculate the mean energy, variance energy, mean absolute value, and variance absolute value for each window.
10)	Continue with the next subband
11)	Continue with the next level
12)	Continue with the next image
13)	Continue with the next sample set
14)	Continue with the next defect type
15)	Continue with the next fabric type

6.6.2 Feature extraction algorithm for DTCWT

The algorithm for DTCWT feature extraction is similar to the UDWT feature extraction program and therefore we use the same pseudo code in Table 6-3 to describe it. The algorithm was performed for each of 6 wavelets designed under Section 6.4 because the results were used to compare the performance of the different wavelets.

In step 5) we perform the five level DTCWT decomposition of the image, getting six subbands with complex coefficients for each of the five levels of decompositions. The six subbands corresponds to the six orientations -75° , -45° , -15° , $+15^\circ$, $+45^\circ$ and $+75^\circ$.

Step 7) is performed for each of the six subbands rather than three subbands as used for UDWT. In step 8) the size of the window is not 32x32, but rather depends on the level of decomposition as follows: for level 1 the size is 16x16, for level 2 the size is 8x8, for level 3

the size is 4x4, for level 4 the size is 2x2 and for level 5 the size is 1x1. This is due to the downsampling performed during the wavelet decomposition process.

In step 9) the calculation done is for the mean energy, variance of energy, mean magnitude and variance of magnitude of complex wavelet coefficients.

The 2D dual-tree wavelet decomposition in step 5) was performed using the MATLAB function '*cplxduel2*' provided by Polytechnic University, Brooklyn, NY [183]. Given the filters $H_0^{(1)}$, $H_1^{(1)}$, $G_0^{(1)}$, $G_1^{(1)}$, H_0 , H_1 , G_0 and G_1 as they appear in Figure 6-2, this function performs a multilevel 2D dual-tree complex wavelet decomposition of an image giving out the real and imaginary parts of the complex wavelet coefficients of the six channels for each level of decomposition.

The filters H_1 , G_0 and G_1 are obtained from the filters H_0 available from Table 6-2. Filters G_0 are time-reversed versions of the corresponding filters H_0 . Filters H_1 and G_1 are the modulated versions of the corresponding filters H_0 and G_0 respectively. This means that, for instance, to get the coefficients of H_1 from those of H_0 , we keep unchanged the odd-numbered ones while we multiply by -1 the even-numbered ones.

6.7 Compared defect detection performance of DTCWT and UDWT

We performed the defect detection experiments using DTCWT- and UDWT-based features and the MCE-trained Euclidean distance classifier. Figure 6-3 and Table 6-4 show the average defect detection rate of DTCWT- and UDWT-based features for the different wavelets designed in sections 6.3 and 6.4; the average is calculated over the whole dataset. We see that DTCWT features perform better than UDWT features. This was expected as DTCWT can separate features in six different directions, while UDWT can separate features in only three directions. In addition, the separation of features in the third direction (diagonal direction) is not complete, because UDWT is not able to distinguish features oriented parallel to the main diagonal (45°) from those oriented parallel to the secondary diagonal (-45°).

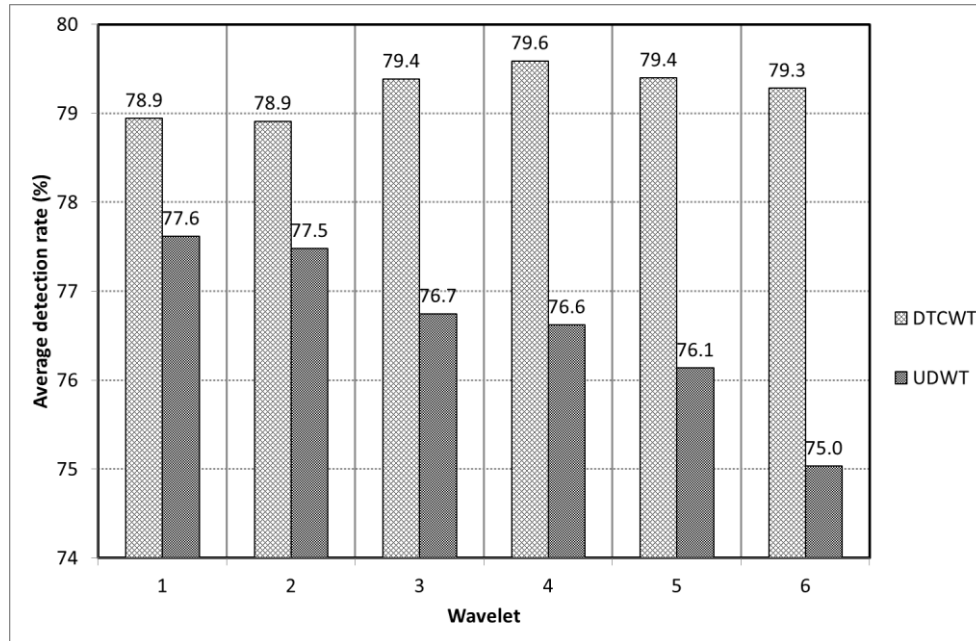


Figure 6-3: Comparison of detection rate of DTCWT- and UDWT-based features as the wavelet filter length increases

From the same results we see that the difference of detection rate between DTCWT- and UDWT-based features increases as the filter length increases. This is due to the observation that the defect detection performance of UDWT-based features generally decreases as the filter length increases, while the defect performance of DTCWT-based features tends to increase.

Table 6-4: Average detection rate (%) of DTCWT- and UDWT-based features as the wavelet filter length increases

Wavelet	Filter length	DTCWT	UDWT
1	4	78.9	77.6
2	6	78.9	77.5
3	8	79.4	76.7
4	10	79.6	76.6
5	12	79.4	76.1
6	14	79.3	75.0

Figure 6-4 and Table 6-5 compare the defect detection performance of the DTCWT- and UDWT-based features for the different types of fabrics. Except for the fabrics of types C4R1 and C4R3, the DTCWT-based features outperform the UDWT-based features. For these two types of fabrics the UDWT-based features produce much better defect detection rates than DTCWT-based features. One possible explanation is that, unlike the other fabric types in our

dataset, the fabric types C4R1 and C4R3 have texture with no apparent periodicity. The superiority of the DTCWT-based features over the UDWT-based features lies in their ability to represent texture in different orientations as explained earlier in this section. The fabric types C4R1 and C4R3 have no apparent periodicity and therefore low regularity in different directions. That could have resulted in the loss of the competitive attribute of the DTCWT-features over the UDWT-based features to the point that the latter performed better for those specific fabric types.

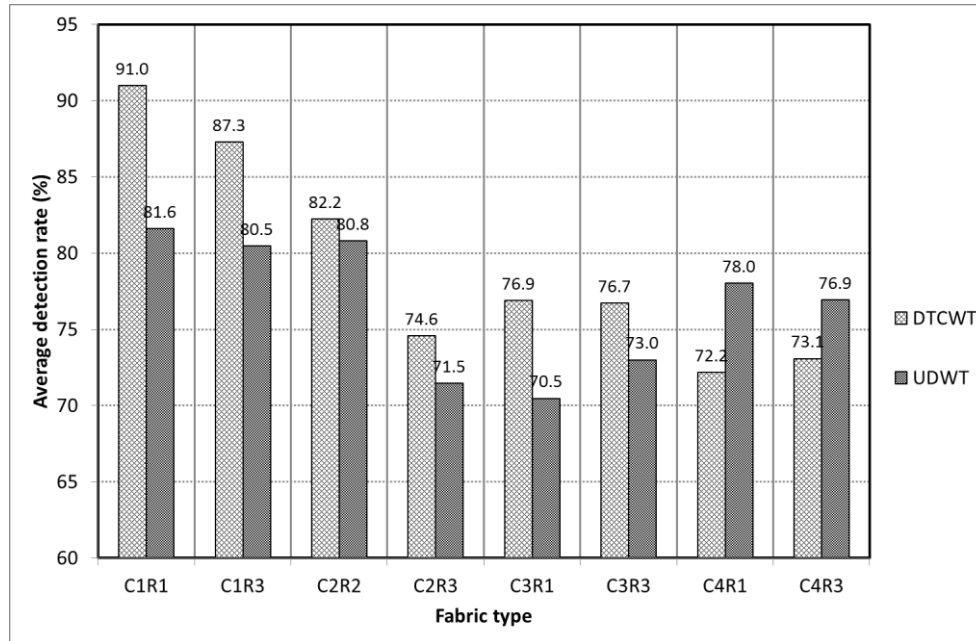


Figure 6-4: Comparison of detection rate of DTCWT- and UDWT-based features for the eight types of fabrics

Figure 6-5 shows a detailed graph of the comparison for fabric types C4R1 and C4R3. It shows the variation of the fabric defect detection rate as the wavelet size of support changes for both UDWT- and DTCWT- based features. From these results it can be seen that the best detection performance for fabric type C4R1 is achieved using UDWT wavelet 3 (which uses filters of length 6), while for the fabric type C4R3 the best performance is achieved using UDWT wavelet 2 (filter size=4).

Table 6-5: Average detection rate (%) of DTCWT- and UDWT-based features for the eight fabric types

Fabric type	DTCWT	UDWT
C1R1	91.0	81.6
C1R3	87.3	80.5
C2R2	82.2	80.8
C2R3	74.6	71.5
C3R1	76.9	70.5
C3R3	76.7	73.0
C4R1	72.2	78.0
C4R3	73.1	76.9

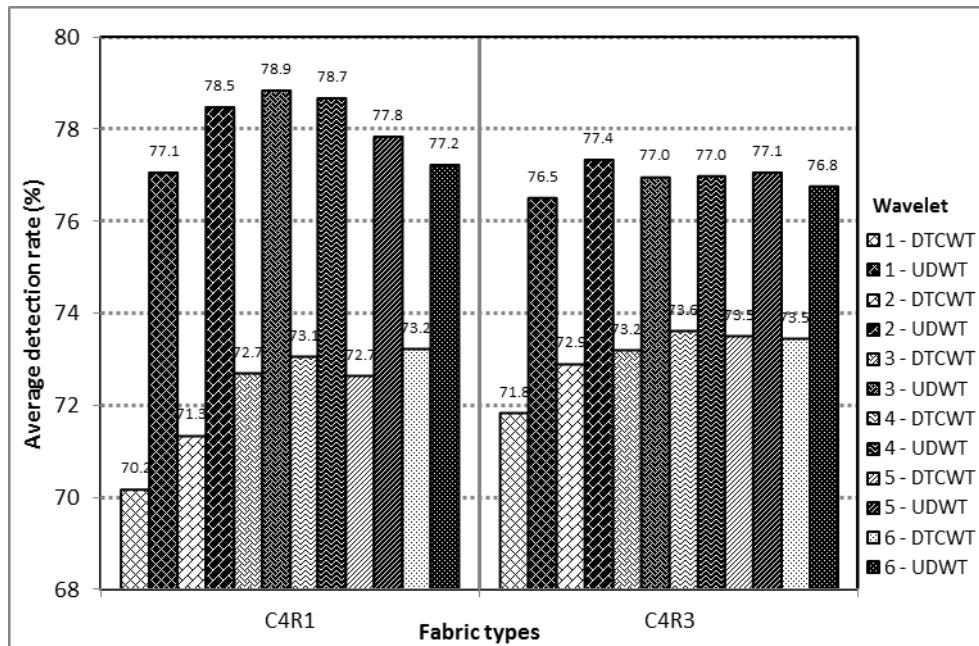


Figure 6-5: Comparison of detection performance of DTCWT- and UDWT-based features for the fabrics of types C4R1 and C4R3 for different wavelet support sizes

6.8 Defect detection performance of wavelet features vs the wavelet size of support

6.8.1 DTCWT-based features

Figure 6-6 and Table 6-6 show the results of detection of DTCWT-based features extracted using wavelets of different support sizes and for the eight fabric types. Two observations can be made from those results.

First, the detection performance does not depend much on the size of support of the wavelet. For all the wavelets used with filters sized from 4 to 14, and for all the eight fabric types, the highest difference in detection rate is 2.7% observed for the fabric type C4R1.

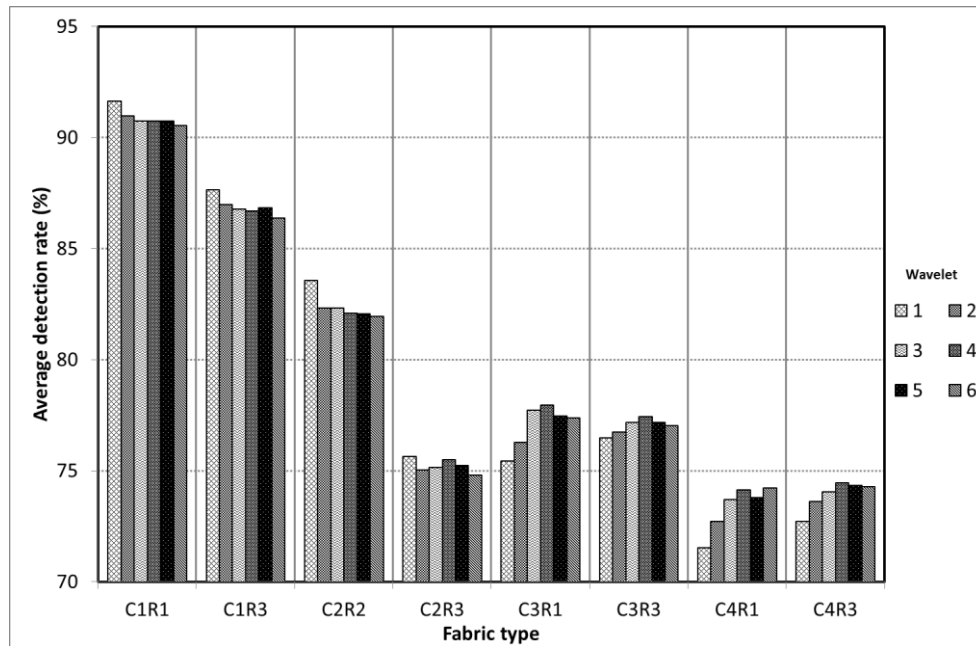


Figure 6-6: Detection rate of DTCWT-based features as the wavelet filter length increases for the different types of fabric.

Second, that small variation is such that short wavelets tend to perform better than long wavelet for fabrics with fine texture (C1R1, C1R3, C2R2 and C2R3), while long wavelets tend to perform better than short wavelets for fabrics with coarser texture (C3R1, C3R3, C4R1 and C4R3). For all these fabrics with fine texture the best detection performance is achieved by wavelet 1, which uses filters of length 4. For all these fabrics with coarser texture, the best detection performance is achieved by using wavelet 4, which uses filters of length 10.

Table 6-6: Average detection rate (%) of DTCWT-based features as the wavelet filter length increases for the eight types of fabrics.

Fabric type	Wavelet	1	2	3	4	5	6
	Filter length	4	6	8	10	12	14
C1R1		91.6	91.0	90.7	90.7	90.8	90.5
C1R3		87.7	87.0	86.8	86.7	86.8	86.4
C2R2		83.6	82.3	82.3	82.1	82.1	81.9
C2R3		75.6	75.0	75.1	75.5	75.2	74.8
C3R1		75.4	76.3	77.7	78.0	77.5	77.4
C3R3		76.5	76.8	77.2	77.4	77.2	77.0
C4R1		71.5	72.7	73.7	74.1	73.8	74.2
C4R3		72.7	73.6	74.0	74.5	74.3	74.3

6.8.2 UDWT-based features

Figure 6-7 and Table 6-7 show the defect detection results of UDWT-based features for wavelets of different support sizes and for all eight types of fabrics present in our dataset. The following observations can be made.

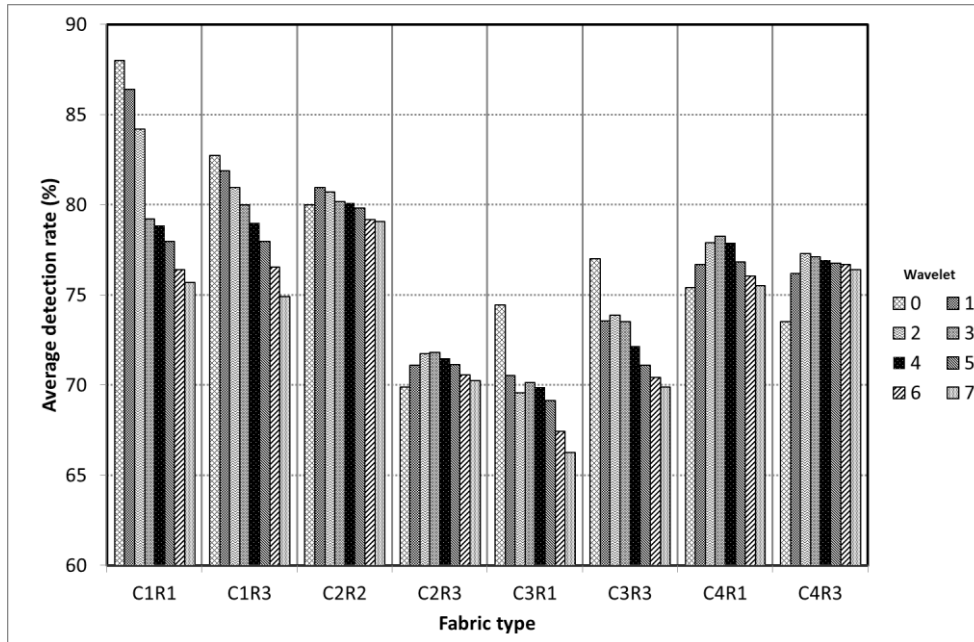


Figure 6-7: Detection rate of UDWT-based features as the wavelet filter length increases for the eight different types of fabric.

For very fine fabrics (C1R1 and C1R3) and for fabrics with a periodic structure (C3R1 and C3R3) the detection rate decreases as the wavelet length increases. For all those cases, the best detection rate is observed for wavelet 0, which uses filters of length 2. Also, for these types of fabrics the variation of the detection rate with respect to the size of wavelet support is

great. For example, for fabric type C1R1 there is a difference on detection rate of 12.3% between the detection rate of 88.0% achieved using the shortest wavelet (wavelet 0) and the detection rate of 75.7% obtained using the longest wavelet (wavelet 7). Therefore, for these types of fabrics it is preferable to use wavelet 0, implemented with filters of length 2. This has the added advantage of lower computational load.

Table 6-7: Average detection rate (%) UDWT-based features as the wavelet filter length increases for the 8 types of fabrics

Fabric type	Wavelet	0	1	2	3	4	5	6	7
	Filter length	2	4	6	8	10	12	14	16
C1R1		88.0	86.4	84.2	79.2	78.8	78.0	76.4	75.7
C1R3		82.7	81.9	81.0	80.0	79.0	78.0	76.6	74.9
C2R2		80.0	81.0	80.7	80.2	80.1	79.8	79.2	79.1
C2R3		69.9	71.1	71.8	71.8	71.5	71.2	70.6	70.2
C3R1		74.4	70.5	69.6	70.1	69.9	69.2	67.4	66.3
C3R3		77.0	73.5	73.9	73.5	72.2	71.1	70.4	69.9
C4R1		75.4	76.7	77.9	78.3	77.9	76.8	76.0	75.5
C4R3		73.5	76.2	77.3	77.1	76.9	76.7	76.7	76.4

For fabrics with a low variance structure (C2R2 and C2R3), and for printed fabrics with no apparent periodicity (C4R1 and C4R3), the detection rate slightly increases with the wavelet size of support and then slightly decreases with it. That may be due to the fact that long wavelets involve distant pixels from each other which might be statistically unrelated. However, for these types of fabrics the overall variation in detection rate with respect to the wavelet size of support remains relatively small. The greatest variation of 3.8% is observed for fabric type C4R3 where wavelet 0 achieves 73.5%, while wavelet 2 achieves 77.3%.

6.9 Defect detection performance of wavelet features vs the number of wavelet decomposition levels

In this section we study the variation of detection performance as the number of decomposition levels increases. This will allow us to decide on the appropriate number of wavelet decomposition levels.

Figure 6-8 shows how the average defect rate varies as the number of wavelet decomposition levels increases for both the DTCWT- and UDWT-based features. For DTCWT-based features the average defect detection rate increases significantly with the addition of every decomposition level. This means that all five the decomposition levels are very important.

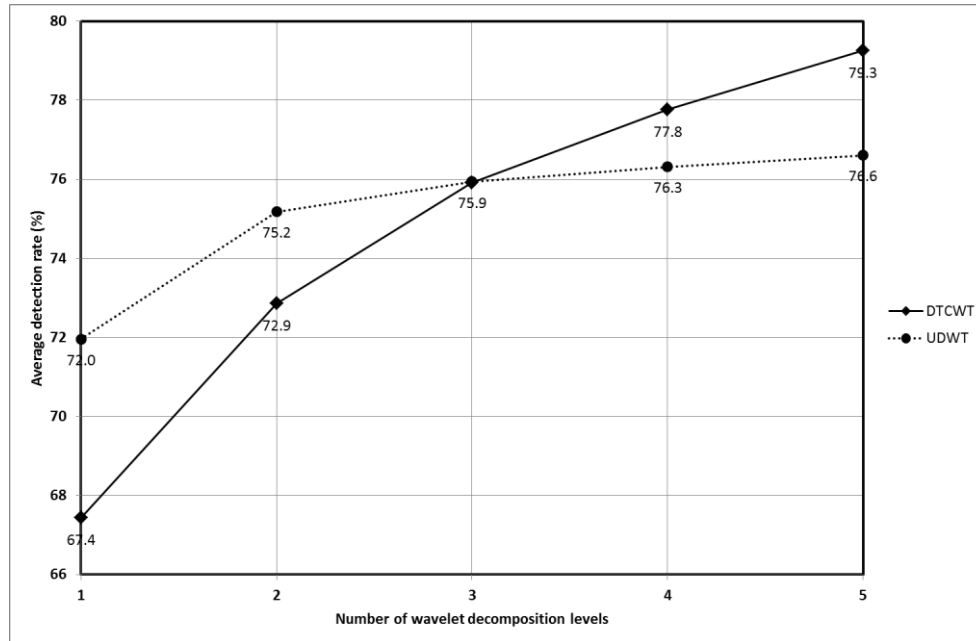


Figure 6-8: Variation of average defect detection rate with respect to the number of wavelet decomposition levels

In contrast, for UDWT-based features there is a significant increase in the average detection rate as the number of wavelet decomposition levels is increased from 1 to 2. After two levels of decomposition, the average defect detection rate continues increasing, but at a very low rate. There is little improvement of the average defect detection rate after three levels of wavelet decomposition.

Particularly for fabric types C4R1 and C4R3, for which the detection performance is better for UDWT than DTCWT, we need to decide on the appropriate level of wavelet decomposition. Figure 6-9 shows that there is little improvement in detection rate by increasing the number of decomposition levels beyond three. As the computational cost for UDWT increases exponentially with the number of decomposition levels, it would be computationally wasteful to go beyond level three in these cases and therefore the selected number of decomposition level is **three**.

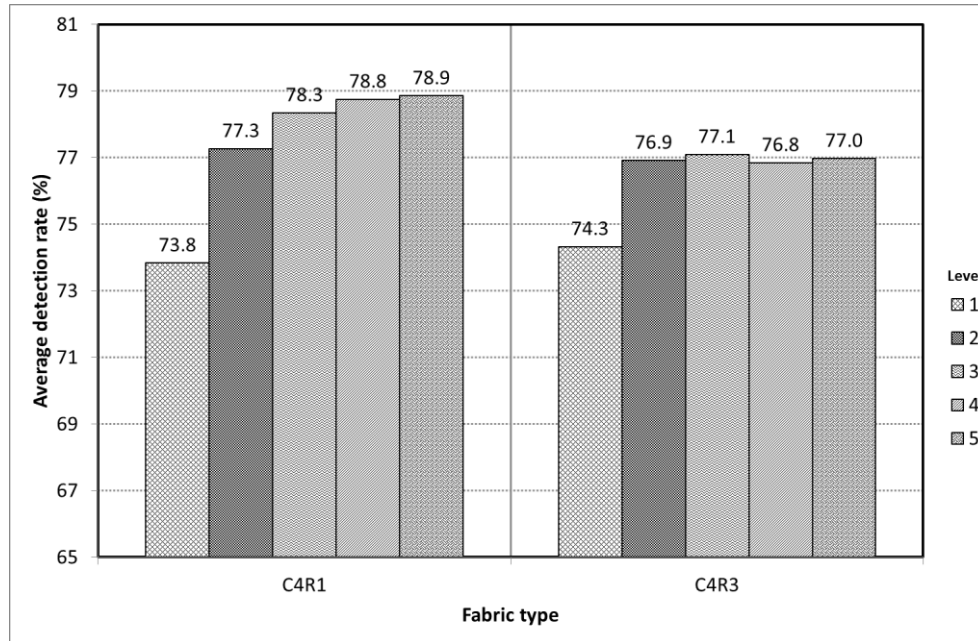


Figure 6-9: Variation of average defect detection rate with respect to the number of UDWT wavelet decomposition levels for fabric types C4R1 and C4R3

6.10 Summary of best wavelet-based features per fabric types

Table 6-8 provides the best wavelet-based features per fabric type obtained from the previous experiments. Remember that the MCE-trained Euclidean distance classifier was used for all the experiments. For fine texture fabrics (C1R1, C1R3, C2R2 and C2R3) these features are the means and variances of magnitudes of DTCWT complex wavelet coefficients (Set c2) obtained using short filters of length 4.

For fabrics with a periodic structure (C3R1 and C3R3), the best wavelet-features obtained are based on DTCWT using filters of length 10. They are means and variances of magnitudes of DTCWT complex wavelet coefficients (Set c2) for C3R1 while they are means and variances of energies of DTCWT complex wavelet coefficients (Set c1) for C3R3.

Finally, for fabrics with no apparent periodicity the best features obtained are based on UDWT using filters of length 8 or 6. The features themselves are means and variances of absolute values of wavelet coefficients (Set r2) for each of wavelet subbands obtained with three wavelet decomposition levels.

Table 6-8: Best wavelet-based features per fabric type

Fabric type	Wavelet transform type	Wavelet filter length	Number of decomposition levels	Feature set	Best average detection rate (%)
C1R1	DTCWT	4	5	Set c2	91.8
C1R3	DTCWT	4	5	Set c2	87.9
C2R2	DTCWT	4	5	Set c2	83.6
C2R3	DTCWT	4	5	Set c1	76.3
C3R1	DTCWT	10	5	Set c2	78.0
C3R3	DTCWT	10	5	Set c1	77.6
C4R1	UDWT	8	3	Set r2	78.3
C4R3	UDWT	8	3	Set r2	77.1

6.11 Effect of the classifier on the performance of wavelet features

The observations made and results obtained in sections 6.7 through 6.10 were based on the use of the MCE-trained Euclidean distance classifier. In this section we investigate whether or not the same observations can be made when other classifiers are used. We will use (i) the K-nearest-neighbour classifier (K=3) and (ii) the feed-forward neural network.

The characteristics of the neural network used were as follows. The number of inputs was 54 for DTCWT-based features and 30 for UDWT-based features. As explained in Subsection 3.5.5, we used one hidden layer with 38 neurons for DTCWT-based features and 22 for UDWT-based features.

As we did for all our defect detection experiments involving a feedforward neural network classifier, two neurons were used in the output layer to represent the defect-free and defective states of the sample under study. The neural network classifier was implemented as described in Subsection 5.6.2 using the Matlab Neural Network Toolbox with the same training parameters and the same strategy to prevent the problem of undertraining.

The main observations we seek to confirm or reject are as follows.

- DTCWT performs better than UDWT for fabric types C1R1, C1R3, C2R2, C2R3, C3R1 and C3R3, while UDWT performs better for fabric types C4R1 and C4R3.
- There is a decreasing trend of detection performance for UDWT and an increasing trend for DTCWT as the wavelet filter length increases when the defect detection rate is averaged across different types of fabrics.

- (c) When DTCWT is used for fine fabrics the shortest wavelet (wavelet 1, filter length=4) performs best, while for coarser fabrics the best detection performance is achieved by using wavelet 4, which uses filters of length 8.
- (d) For most fabric types the shortest wavelet (wavelet 0, filter length=2) performs the best (C1R1, C1R3, C3R1 and C3R3). For the other fabric types wavelets 2 (filter length=6) or wavelet 3 (filter length=8) performs the best with UDWT.
- (e) For DTCWT all the five decomposition levels are useful as they increase significantly the defect detection rate, while for the UDWT three levels of decompositions are enough.

6.11.1 Comparison of UDWT vs DTCWT for different fabric types when the KNN and the neural network classifiers are used

Figure 6-10 and Figure 6-11 show the comparison of the average defect detection rates of DTCWT- and UDWT-based features for the eight fabric types when respectively the K-nearest-neighbour and the feed-forward neural network classifiers are used. From both figures we can see that the average defect detection rate of DTCWT-based features is higher than the UDWT-based features for fabric types C1R1, C1R3, C2R2, C2R3, C3R1 and C3R3, while the reverse is true for fabric types C4R1 and C4R3. Therefore observation (a) remains valid for all three classifiers.

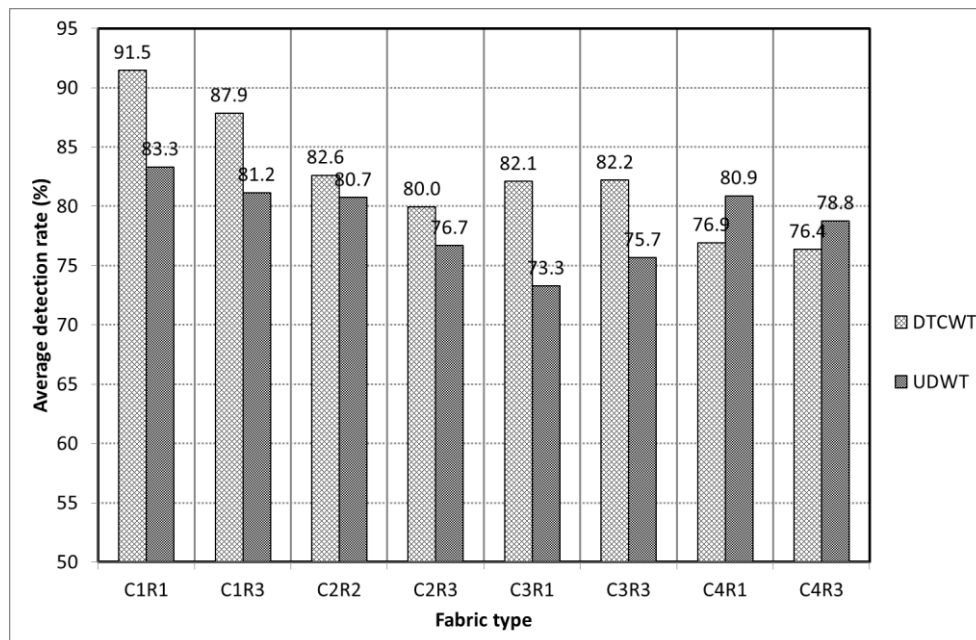


Figure 6-10: Comparison of detection rate of DTCWT- and UDWT-based features for the eight types of fabrics using the K-nearest-neighbour classifier

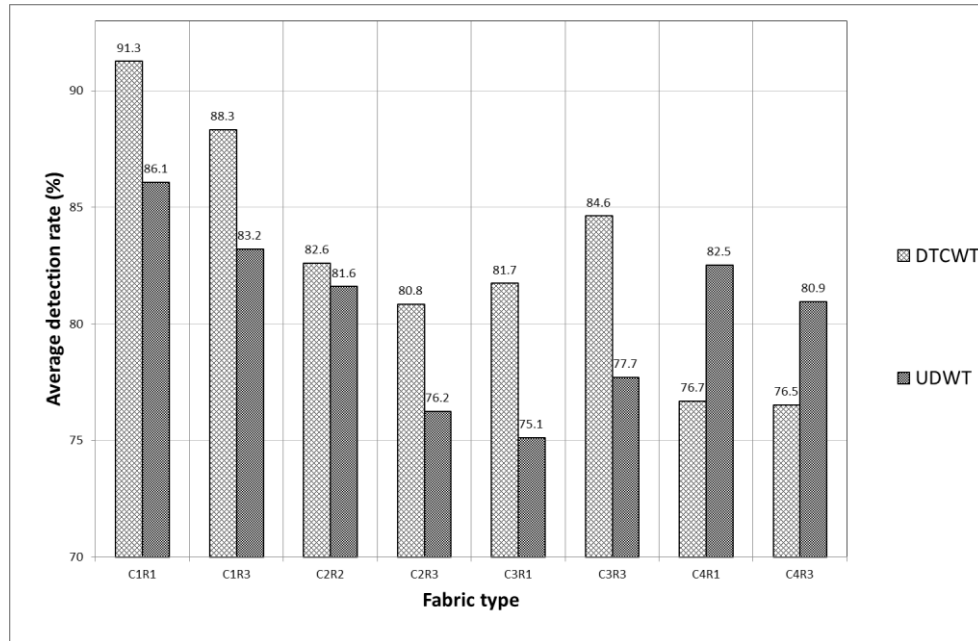


Figure 6-11: Comparison of detection rate of DTCWT- and UDWT-based features for the eight types of fabrics using the feed-forward neural network classifier

6.11.2 Dependence of the defect detection rate on the wavelet size of support when the KNN and the neural network classifiers are used

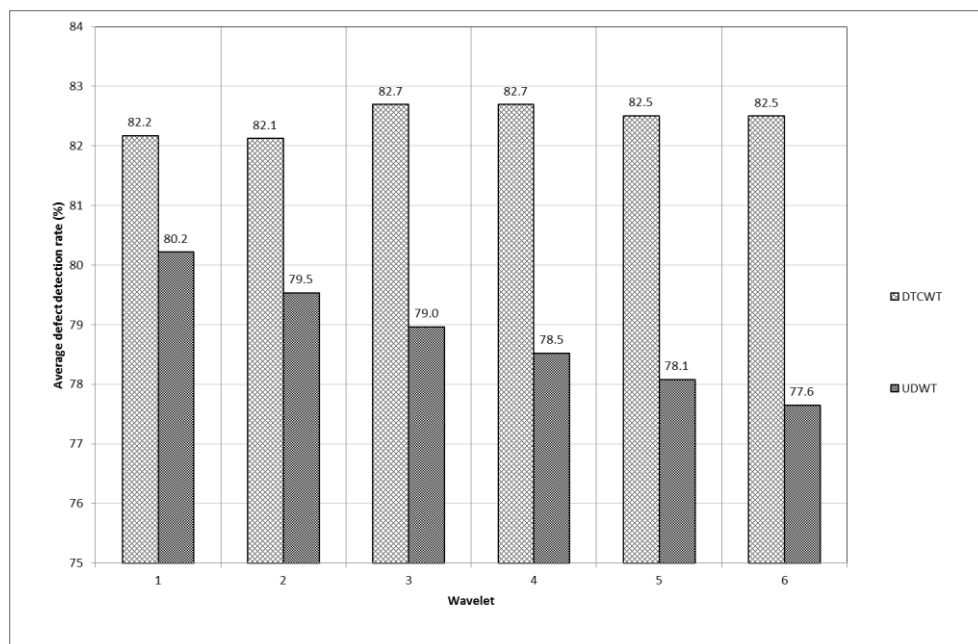


Figure 6-12: Comparison of detection rate of DTCWT- and UDWT-based features as the wavelet filter length increases when a K-nearest-neighbour classifier is used

We can read from both figures 6-12 and 6-13 that the average defect detection rate of UDWT-based features decreases when the wavelet reference number increases whether the K-nearest-neighbour classifier or the feed-forward neural network is used. We can also see from both figures the same slight increase trend of average defect detection rate of the DTCWT-based features as we observed in Figure 6-3 when the MCE-based classifier was used. One is reminded that the wavelet reference number we used increased with the size of support and the wavelet filter length as shown in Tables 6-1 and 6-2. Therefore observation (b) remains valid for all three classifiers.

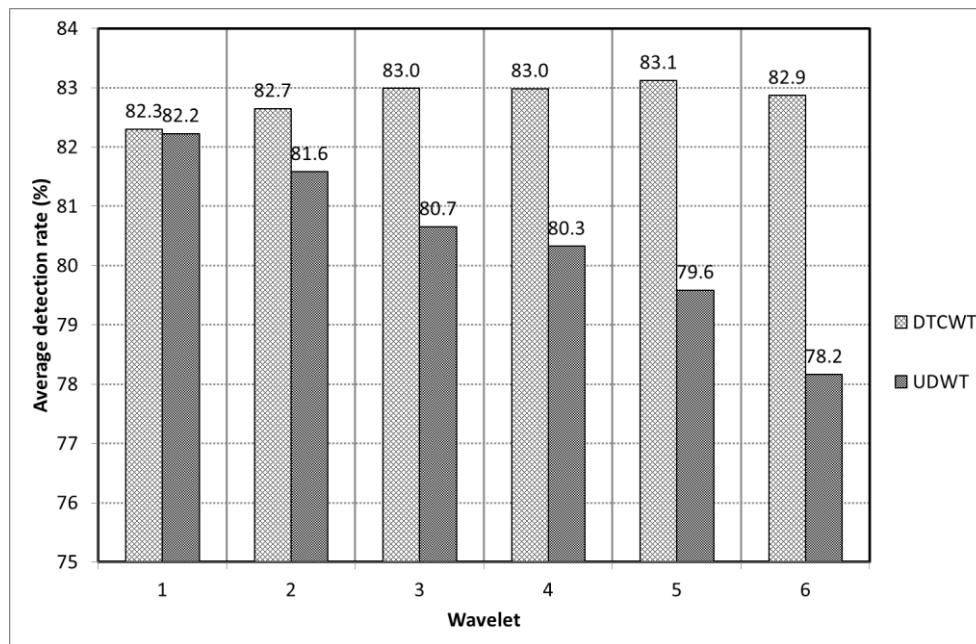


Figure 6-13: Comparison of detection rate of DTCWT- and UDWT-based features as the wavelet filter length increases when a feed-forward neural network classifier is used

6.11.3 Best wavelet filter length for different fabric types when the KNN and the neural network classifiers are used

Figures 6-14 through 6-17 and tables 6-9 through 6-12 show the variation of the average defect detection rates of wavelet-based features both for DTCWT and UDWT when the K-nearest-neighbour and the feed-forward neural network classifiers are used. From these and earlier results for the MCE-trained Euclidean distance classifier (figures 6-6 and 6-7 and tables 6-6 and 6-7) we get the summarised information about the best wavelet filter length for the different fabric types shown in Table 6-13.

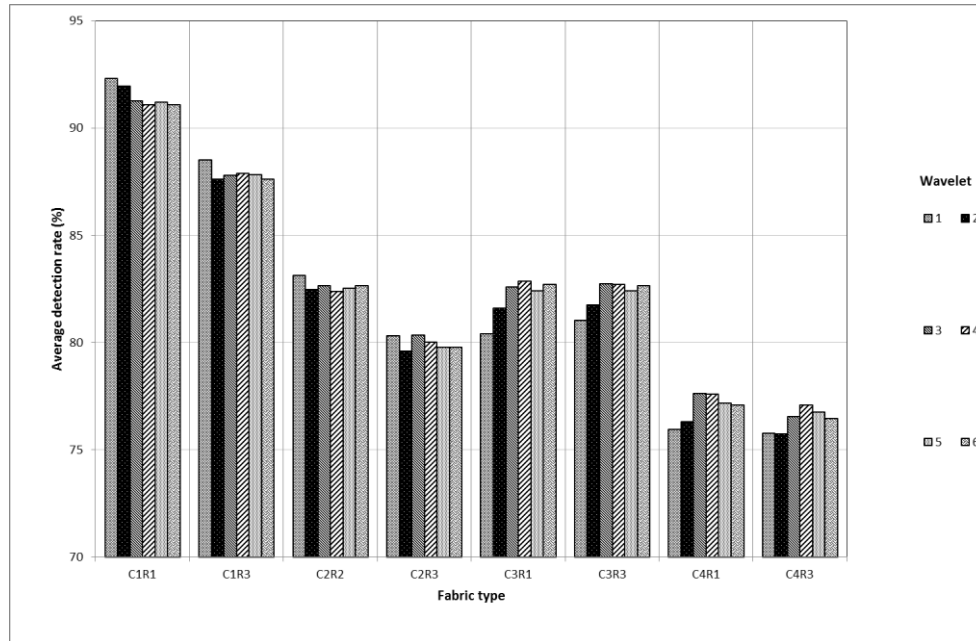


Figure 6-14: Detection rate of DTCWT-based features as the wavelet filter length increases for the different types of fabric when a KNN classifier is used

Table 6-9: Average detection rate (%) of DTCWT-based features as the wavelet filter length increases for the eight types of fabrics when a K-nearest-neighbour classifier is used

Fabric type	Wavelet	1	2	3	4	5	6
	Filter length	4	6	8	10	12	14
C1R1		92.3	91.9	91.3	91.1	91.2	91.1
C1R3		88.5	87.6	87.8	87.9	87.8	87.6
C2R2		83.1	82.5	82.7	82.4	82.5	82.7
C2R3		80.3	79.6	80.3	80.0	79.8	79.8
C3R1		80.4	81.6	82.6	82.9	82.4	82.7
C3R3		81.0	81.8	82.7	82.7	82.4	82.7
C4R1		75.9	76.3	77.6	77.6	77.2	77.1
C4R3		75.8	75.7	76.5	77.1	76.7	76.5

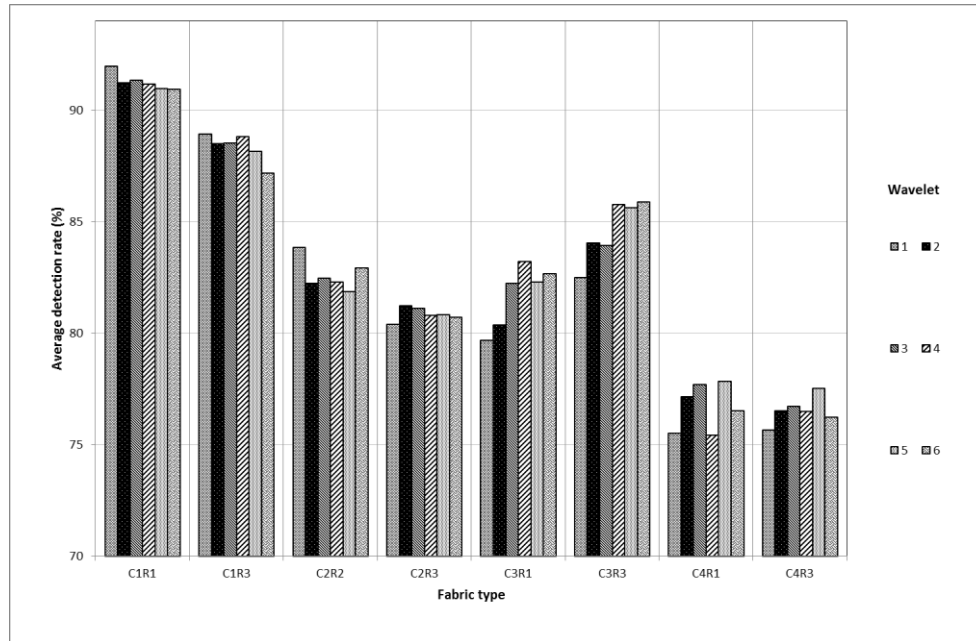


Figure 6-15: Detection rate of DTCWT-based features as the wavelet filter length increases for the different types of fabric when a feed-forward neural network classifier is used

Table 6-10: Average detection rate (%) of DTCWT-based features as the wavelet filter length increases for the eight types of fabrics when a feed-forward neural network classifier is used

Fabric type	Wavelet	1	2	3	4	5	6
	Filter length	4	6	8	10	12	14
C1R1		91.9	91.2	91.3	91.2	91.0	90.9
C1R3		88.9	88.5	88.5	88.8	88.1	87.2
C2R2		83.8	82.2	82.5	82.3	81.8	82.9
C2R3		80.4	81.2	81.1	80.8	80.8	80.7
C3R1		79.7	80.4	82.2	83.2	82.3	82.7
C3R3		82.5	84.0	83.9	85.8	85.6	85.9
C4R1		75.5	77.1	77.7	75.4	77.8	76.5
C4R3		75.6	76.5	76.7	76.5	77.5	76.2

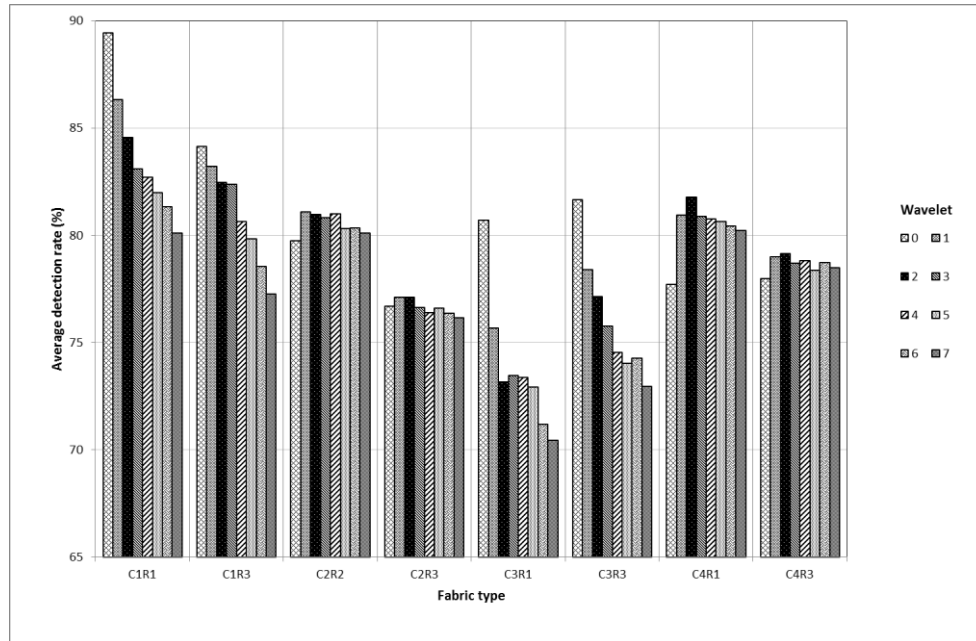


Figure 6-16: Detection rate of UDWT-based features as the wavelet filter length increases for the different types of fabric when a KNN classifier is used

Table 6-11: Average detection rate (%) of UDWT-based features as the wavelet filter length increases for the eight types of fabrics when a K-nearest-neighbour classifier is used

Fabric type	Wavelet	0	1	2	3	4	5	6	7
	Filter length	2	4	6	8	10	12	14	16
C1R1		89.4	86.3	84.6	83.1	82.7	82.0	81.3	80.1
C1R3		84.2	83.2	82.5	82.4	80.6	79.8	78.5	77.2
C2R2		79.7	81.1	81.0	80.8	81.0	80.3	80.3	80.1
C2R3		76.7	77.1	77.1	76.6	76.4	76.6	76.4	76.2
C3R1		80.7	75.7	73.1	73.5	73.4	72.9	71.2	70.4
C3R3		81.6	78.4	77.1	75.8	74.5	74.0	74.3	73.0
C4R1		77.7	80.9	81.8	80.9	80.8	80.6	80.4	80.2
C4R3		78.0	79.0	79.2	78.7	78.8	78.4	78.7	78.5

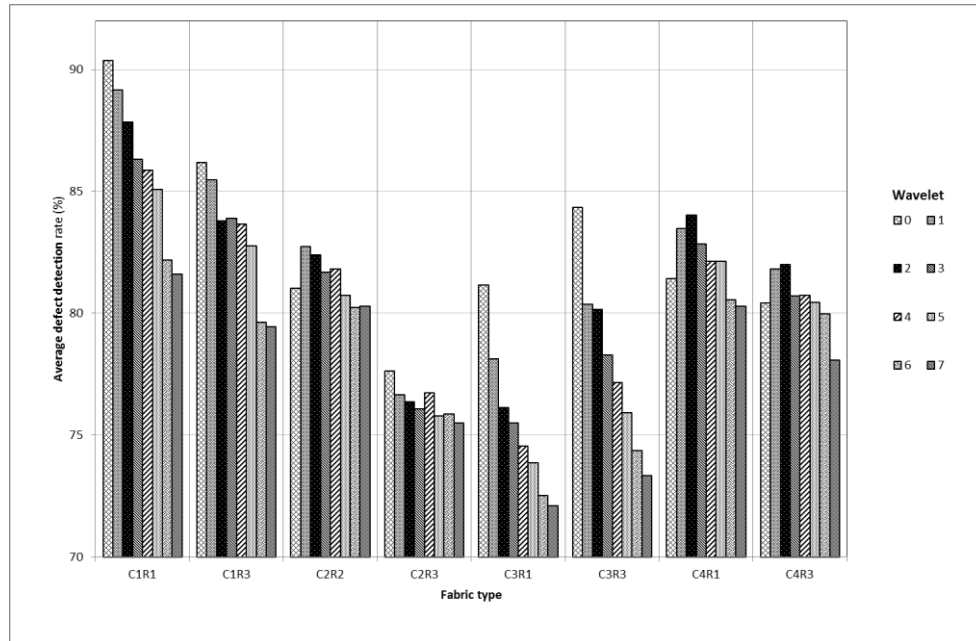


Figure 6-17: Detection rate of UDWT-based features as the wavelet filter length increases for the different types of fabric when a feed-forward neural network classifier is used

Table 6-12: Average detection rate (%) of UDWT-based features as the wavelet filter length increases for the eight types of fabrics when the feed-forward neural network classifier is used

Fabric type	Wavelet	0	1	2	3	4	5	6	7
	Filter length	2	4	6	8	10	12	14	16
C1R1		90.4	89.2	87.8	86.3	85.9	85.1	82.2	81.6
C1R3		86.2	85.5	83.8	83.9	83.7	82.8	79.6	79.4
C2R2		81.0	82.7	82.4	81.7	81.8	80.7	80.2	80.3
C2R3		77.6	76.7	76.4	76.1	76.7	75.8	75.9	75.5
C3R1		81.1	78.1	76.1	75.5	74.5	73.9	72.5	72.1
C3R3		84.3	80.4	80.2	78.3	77.2	75.9	74.4	73.3
C4R1		81.4	83.5	84.0	82.8	82.1	82.1	80.6	80.3
C4R3		80.4	81.8	82.0	80.7	80.7	80.4	80.0	78.1

From Table 6-13 we can see that for the UDWT the shortest wavelet (wavelet 0, filter length=2) performs the best for the most types of fabrics (C1R1, C1R2, C3R1 and C3R3), while for the remaining fabrics types wavelet 2 (filter length = 6) or wavelet 3 (filter length = 8) perform the best. This is true when any of the three classifiers is used and confirms observation (d) when the K-nearest-neighbour and the neural network classifiers are used.

From the same table we also see that for the DTCWT the shortest wavelet used (wavelet 1, filter length = 4) performs best for fine fabrics (C1R1, C1R3, C2R2 and C2R3), while for the coarser fabrics wavelet 4 (with filter length = 10) performs the best. This is also true when any of the three classifiers is used. The exception is when the neural network classifier is used for fabric types C2R3, C4R1 and C4R3, where respectively wavelet 2 (filter length = 6), wavelet 3 (filter length = 8) and wavelet 5 (filter length = 12) perform best. This confirms that observation (c) remains valid when the K-nearest-neighbour and the neural network classifiers are used.

Table 6-13: Best wavelet-filter length when the three different classifiers are used (MCE refers to MCE-trained Euclidean distance classifier, KNN refers to the K-nearest-neighbour classifier, while Neural refers to the feed-forward neural network classifier)

Fabric type	UDWT			DTCWT		
	MCE	KNN	Neural	MCE	KNN	Neural
C1R1	2	2	2	4	4	4
C1R3	2	2	2	4	4	4
C2R2	6	4	4	4	4	4
C2R3	8	4	2	4	4	6
C3R1	2	2	2	10	10	10
C3R3	2	2	2	10	10	10
C4R1	8	6	6	10	10	8
C4R3	6	6	6	10	10	12

6.11.4 Defect detection performance of wavelet-based features with respect to the number of wavelet decomposition levels when the KNN and the neural network classifiers are used

Figures 6-18 and 6-19 show the variation of the average detection rate as the number of wavelet decomposition levels is increased when respectively the K-nearest-neighbour (K=3) and the feed-forward neural network classifiers are used. We can see from both figures that for DTCWT the average detection rate increases significantly after each level of decomposition is added, although at a lower rate after two levels. However, for the UDWT the average detection rate increase is very little after the third level of decomposition and does not justify the additional computational cost. It should be noted that the amount of data to be processed doubles from one level to the next and therefore it is these last decomposition levels that are computationally costly. Therefore, and in agreement with observation (e), all

the five DTCWT wavelet decompositions are required, while only three levels of decomposition levels are required for UDWT.

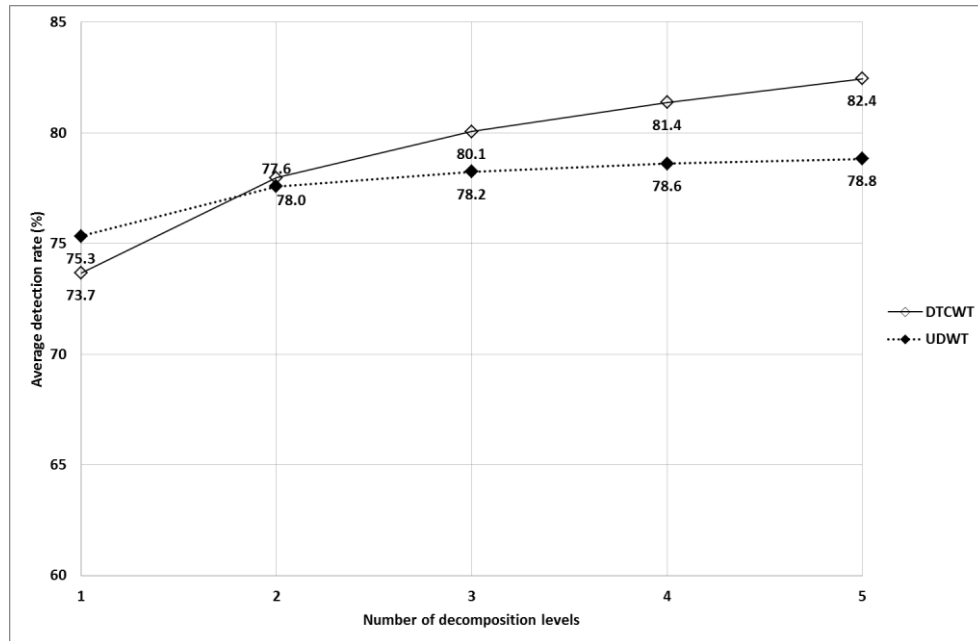


Figure 6-18: Variation of average defect detection rate with respect to the number of wavelet decomposition levels when a K-nearest-neighbour classifier is used

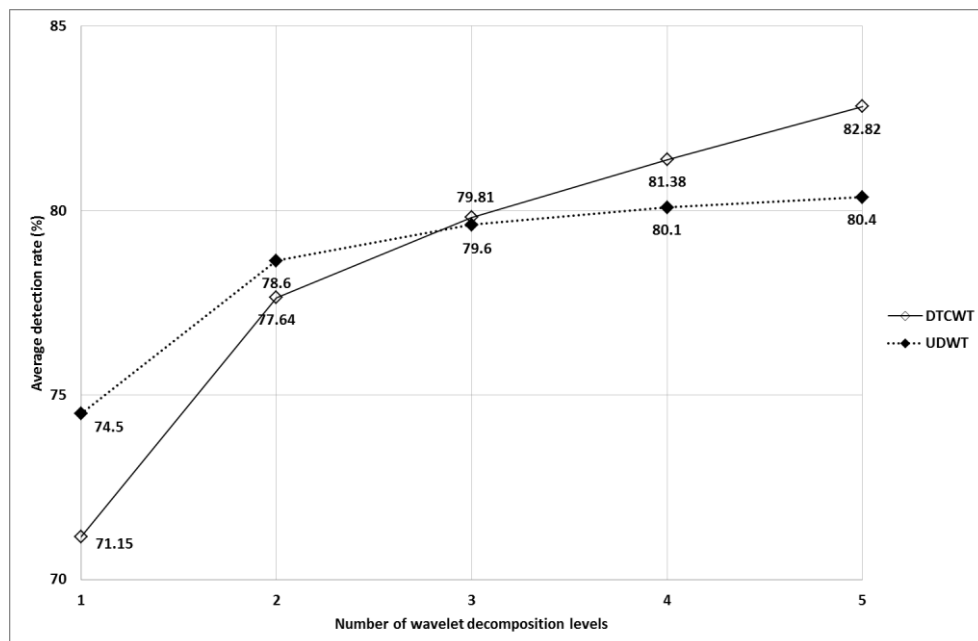


Figure 6-19: Variation of average defect detection rate with respect to the number of wavelet decomposition levels when a feed-forward neural network classifier is used

6.11.5 Summarised comparison of the best-based wavelet features per fabric type for the three classifiers

Table 6-14 summarises the best wavelet features as shown by their defect detection performance by using the three different classifiers.

The detection rates from Table 6-14 are shown graphically in Figure 6-20 and allow us to compare the classifiers' ability to detect fabric defects from wavelet-based features. We can see from the figure that the feed-forward neural network classifier performs the best among the three types of classifiers we used.

Table 6-14: Best wavelet filter features for different fabrics using the three classifiers (BAVR (%) refers to best average detection rate (%))

Fabric type	Classifier					
	MCE-trained Euclidean distance classifier		K-nearest-neighbour classifier		Feed-forward neural network classifier	
C1R1 (DTCWT)	Filter length	4	Filter length	4	Filter length	4
	Feature set	Set c2	Feature set	Set c2	Feature set	Set c2
	BAVR (%)	91.8	BAVR (%)	92.3	BAVR (%)	91.9
C1R3 (DTCWT)	Filter length	4	Filter length	4	Filter length	4
	Feature set	Set c2	Feature set	Set c2	Feature set	Set c2
	BAVR (%)	87.9	BAVR (%)	88.5	BAVR (%)	88.9
C2R2 (DTCWT)	Filter length	4	Filter length	4	Filter length	4
	Feature set	Set c2	Feature set	Set c2	Feature set	Set c2
	BAVR (%)	83.6	BAVR (%)	83.1	BAVR (%)	83.8
C2R3 (DTCWT)	Filter length	4	Filter length	4	Filter length	6
	Feature set	Set c1	Feature set	Set c2	Feature set	Set c2

Fabric type	Classifier					
	MCE-trained Euclidean distance classifier		K-nearest-neighbour classifier		Feed-forward neural network classifier	
	BAVR (%)	76.3	BAVR (%)	80.3	BAVR (%)	81.2
C3R1 (DTCWT)	Filter length	10	Filter length	10	Filter length	10
	Feature set	Set c2	Feature set	Set c2	Feature set	Set c2
	BAVR (%)	78.0	BAVR (%)	83.0	BAVR (%)	83.2
C3R3 (DTCWT)	Filter length	10	Filter length	10	Filter length	10
	Feature set	Set c1	Feature set	Set c2	Feature set	Set c2
	BAVR (%)	77.5	BAVR (%)	82.7	BAVR (%)	85.7
C4R1 (UDWT)	Filter length	8	Filter length	6	Filter length	6
	Feature set	Set r2	Feature set	Set r2	Feature set	Set r2
	BAVR (%)	78.3	BAVR (%)	81.1	BAVR (%)	82.7
C4R3 (UDWT)	Filter length	8	Filter length	6	Filter length	6
	Feature set	Set r2	Feature set	Set r2	Feature set	Set r2
	BAVR (%)	77.1	BAVR (%)	79.8	BAVR (%)	80.7

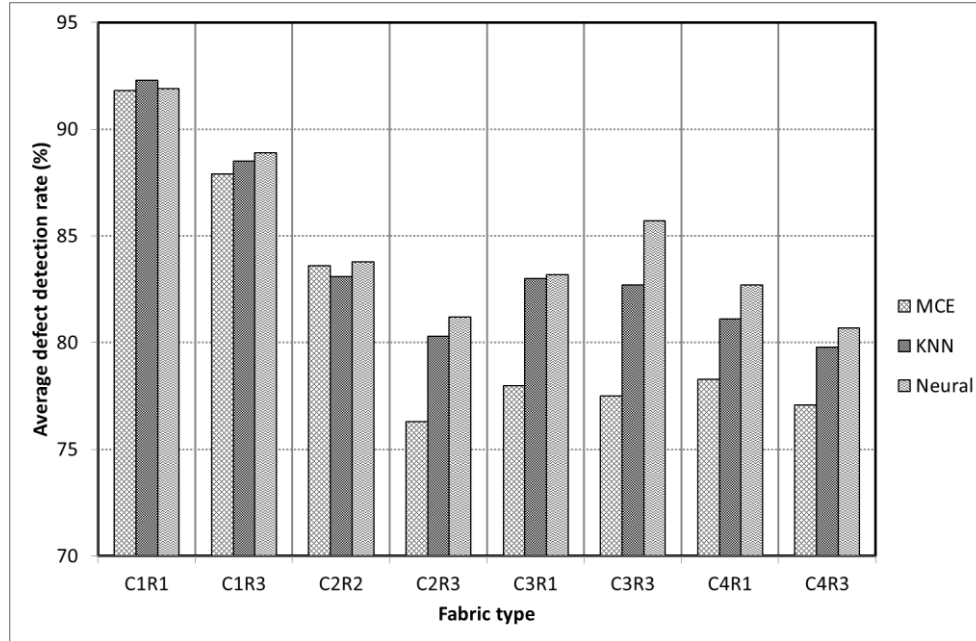


Figure 6-20: Comparison of the classifiers about their defect detection capability from wavelet-based features

6.12 Fabric defect classification using wavelet features

In this section we investigate the variation of the defect classification rate for different types of fabrics when the size of support of the wavelet changes. The objective is to identify for each fabric type the wavelet that provides the features that yield the highest classification rate. We use five wavelet decomposition levels for DTCWT and three wavelet decomposition levels for UDWT as found during the defect detection related experiments in Section 6.9.

The defect classification is done in two stages, as shown earlier in Figure 5-18, with the difference that in this case we use wavelet-based features rather than GLCM features. The first stage is the defect detection stage to discriminate defective from defect-less samples. The samples identified as defective are then submitted to the second classification stage to assign each of them to one of the four types of defects.

Three different classifiers are used: (i) the MCE-trained Euclidean distance classifier, (ii) the K-nearest-neighbour classifier and (iii) the feed-forward neural network classifier. Pertaining to K-nearest-neighbour classifier, we used K=3 for implementing the defect detection stage, and K=9 for the defect classification stage. For the neural network classifier we used the topology shown in Table 6-15, depending on the stage in the classification process and the type of wavelet features.

Table 6-15: Topology of the neural network classifier used for wavelet-based defect classification

		Number of inputs	Number of neurons in the hidden layer	Number of output neurons
Defect detection stage	UDWT-based features	30	22	2
	DTCWT-based features	54	38	2
Defect classification stage	UDWT-based features	30	24	4
	DTCWT-based features	54	40	4

The training parameters and procedure for the neural network classifier are explained in Section 6.11.

6.12.1 Defect classification results of UDWT-based features

Figures 6-21 through 6-23 as well as tables 6-16 through 6-18 show the variation of the defect classification rates obtained from UDWT-based features for different types of fabrics with the increasing wavelet size of support (or the corresponding filter length) when respectively the MCE-trained Euclidean distance classifier, the K-nearest-neighbour classifier and the feed-forward neural network classifiers are used.

The defect classification rates obtained from the MCE-trained Euclidean distance classifier (Figure 6-21) have similar variation trends as those obtained from the K-nearest-neighbour classifier (Figure 6-22). However, the defect classification rates obtained from the K-nearest-neighbour classifier are higher. Two observations require attention: (i) there is a decreasing trend of classification rate as the wavelet size of support increases, apart for the printed fabrics with no apparent periodicity (C4R1 and C4R3) where that trend is less visible; and (ii) the defect classification rates for the printed fabrics with no apparent periodicity (C4R1 and C4R3) are as good as those for the other types of fabrics. This contrasts with the defect detection rates which were consistently lower than those of the other fabric types.

When the feed-forward neural network classifier is used (Figure 6-23), there is a general decreasing trend of the defect classification rates as the wavelet size of support increases for all the types of fabrics in our dataset. The defect classification rates are in the same range as those obtained using the K-nearest-neighbour classifier.

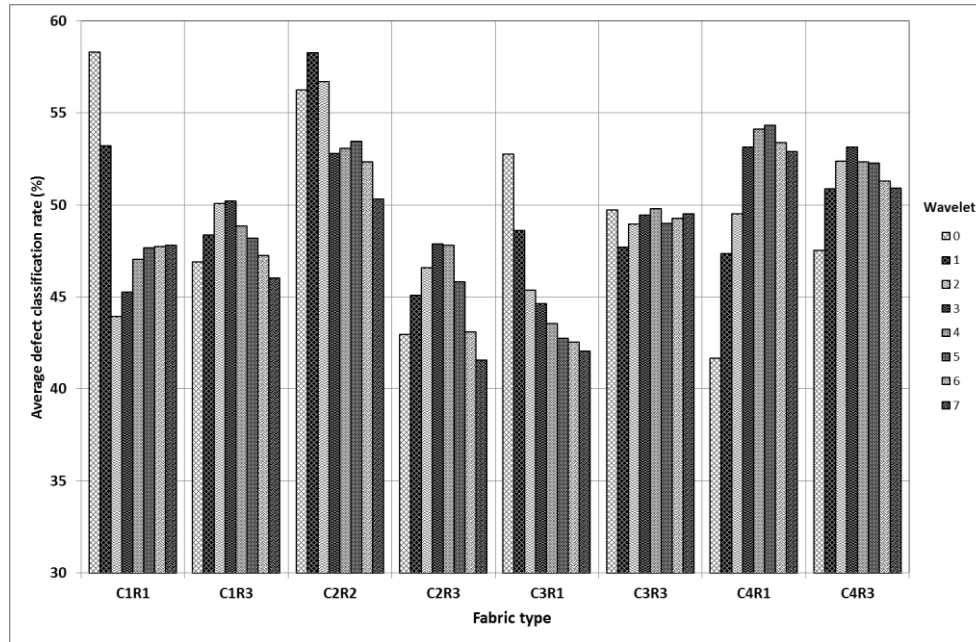


Figure 6-21: Detection classification rate of UDWT-based features as the wavelet filter length increases for the different types of fabric when a MCE-trained Euclidean distance classifier is used

Table 6-16: Average defect classification rate (%) of UDWT-based features as the wavelet filter length increases for the eight types of fabrics when a MCE-trained Euclidean distance classifier is used

Fabric type	Wavelet	0	1	2	3	4	5	6	7
	Filter length	2	4	6	8	10	12	14	16
C1R1		58.3	53.2	43.9	45.3	47.0	47.7	47.7	47.8
C1R3		46.9	48.4	50.1	50.2	48.9	48.2	47.3	46.0
C2R2		56.3	58.3	56.7	52.8	53.1	53.5	52.4	50.3
C2R3		43.0	45.1	46.6	47.9	47.8	45.8	43.1	41.6
C3R1		52.8	48.6	45.4	44.6	43.6	42.8	42.5	42.1
C3R3		49.7	47.7	49.0	49.5	49.8	49.0	49.3	49.5
C4R1		41.7	47.4	49.5	53.2	54.1	54.4	53.4	52.9
C4R3		47.6	50.9	52.4	53.2	52.4	52.3	51.3	50.9

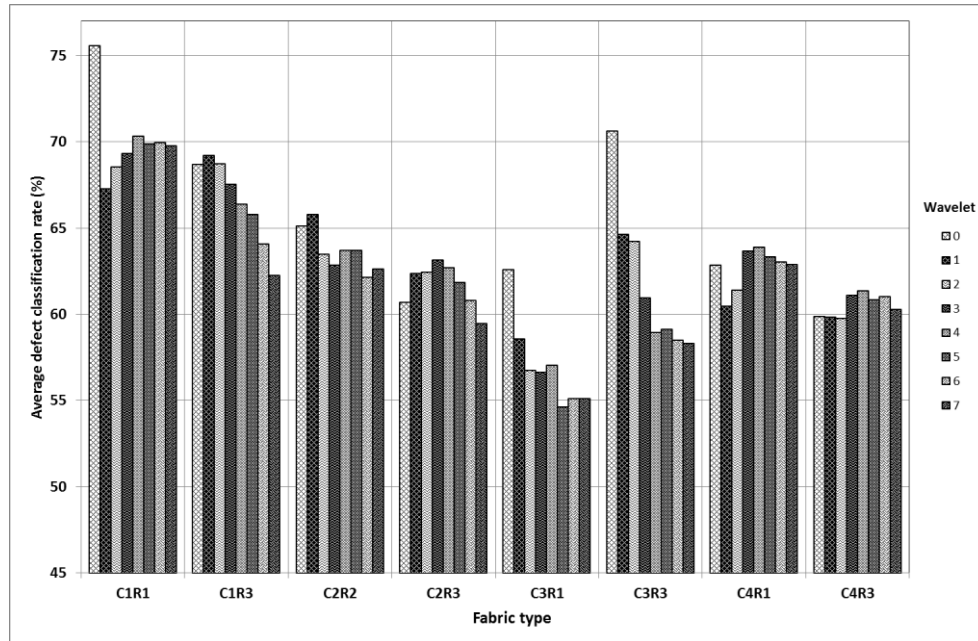


Figure 6-22: Detection classification rate of UDWT-based features as the wavelet filter length increases for the different types of fabric when a K-nearest-neighbour classifier is used

Table 6-17: Average defect classification rate (%) of UDWT-based features as the wavelet filter length increases for the eight types of fabrics when a K-nearest-neighbour classifier is used

Fabric type	Wavelet	0	1	2	3	4	5	6	7
	Filter length	2	4	6	8	10	12	14	16
C1R1		75.6	67.3	68.6	69.3	70.3	69.9	70.0	69.8
C1R3		68.7	69.2	68.7	67.6	66.4	65.8	64.1	62.3
C2R2		65.1	65.8	63.5	62.8	63.7	63.7	62.2	62.6
C2R3		60.7	62.4	62.5	63.2	62.7	61.8	60.8	59.5
C3R1		62.6	58.6	56.7	56.7	57.0	54.6	55.1	55.1
C3R3		70.6	64.7	64.2	61.0	59.0	59.1	58.5	58.3
C4R1		62.9	60.5	61.4	63.7	63.9	63.4	63.1	62.9
C4R3		59.9	59.8	59.8	61.1	61.4	60.8	61.0	60.3

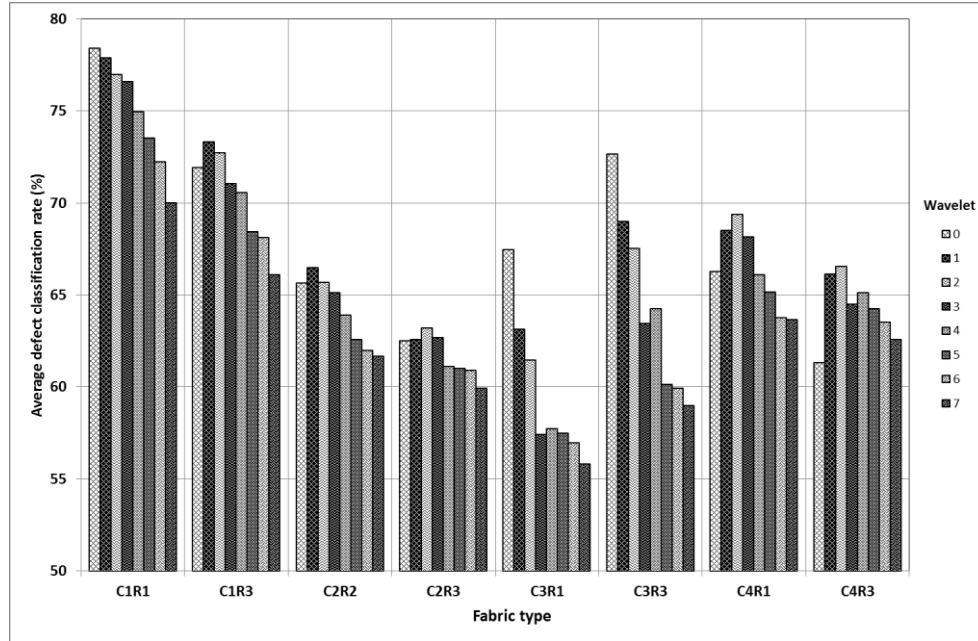


Figure 6-23: Detection classification rate of UDWT-based features as the wavelet filter length increases for the different types of fabric when a feed-forward neural network classifier is used

Table 6-18: Average defect classification rate (%) of UDWT-based features as the wavelet filter length increases for the eight types of fabrics when a feed-forward neural network classifier is used

Fabric type	Wavelet	0	1	2	3	4	5	6	7
	Filter length	2	4	6	8	10	12	14	16
C1R1		78.4	77.9	77.0	76.6	75.0	73.6	72.3	70.0
C1R3		71.9	73.3	72.8	71.1	70.6	68.4	68.1	66.1
C2R2		65.7	66.5	65.7	65.1	63.9	62.6	62.0	61.7
C2R3		62.5	62.6	63.2	62.7	61.1	61.0	60.9	59.9
C3R1		67.5	63.1	61.5	57.4	57.7	57.5	57.0	55.8
C3R3		72.7	69.0	67.5	63.5	64.3	60.1	59.9	59.0
C4R1		66.3	68.5	69.4	68.2	66.1	65.2	63.8	63.7
C4R3		61.3	66.2	66.6	64.5	65.1	64.3	63.5	62.6

6.12.2 Defect classification results of DTCWT-based features

Figures 6-24 through 6-26 and tables 6-19 through 6-21 show the variation of the defect classification rates obtained from DTCWT-based features for different types of fabrics with the increasing wavelet size of support (or the corresponding filter length) when respectively

the MCE-trained Euclidean distance classifier, the K-nearest-neighbour classifier and the feed-forward neural network classifiers are used.

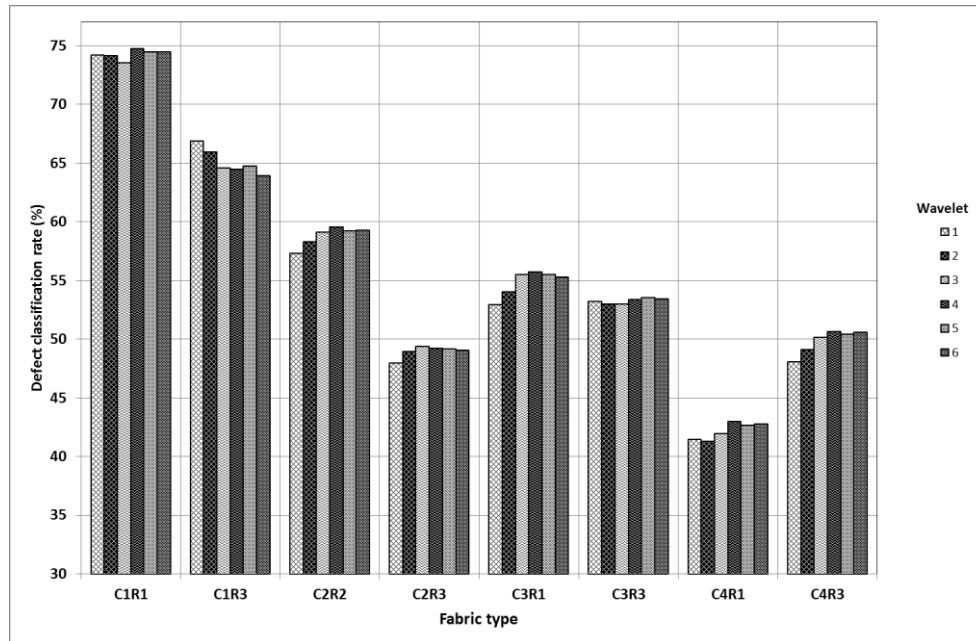


Figure 6-24: Detection classification rate of DTCWT-based features as the wavelet filter length increases for the different types of fabric when MCE-trained Euclidean distance classifier is used

Table 6-19: Average defect classification rate (%) of DTCWT-based features as the wavelet filter length increases for the eight types of fabrics when MCE-trained Euclidean distance classifier is used

Fabric type	Wavelet	1	2	3	4	5	6
	Filter length	4	6	8	10	12	14
C1R1		74.2	74.1	73.5	74.7	74.5	74.5
C1R3		66.9	66.0	64.6	64.5	64.7	64.0
C2R2		57.3	58.3	59.1	59.6	59.3	59.3
C2R3		48.0	49.0	49.4	49.3	49.2	49.1
C3R1		52.9	54.0	55.5	55.7	55.5	55.3
C3R3		53.2	53.0	53.0	53.4	53.6	53.4
C4R1		41.5	41.3	42.0	43.0	42.7	42.8
C4R3		48.1	49.1	50.2	50.6	50.4	50.6

Generally, there is little variation in the defect classification rates with respect to the size of support of the wavelet. Exceptions are seen for fabrics with a clearly visible periodic structure (C3R1 and C3R3) for which a clearly visible increasing trend is observed when the K-

nearest-neighbour and the feed-forward neural network classifiers are used. To a lesser degree a similar trend is observed for the printed fabrics with no apparent periodicity (C4R1 and C4R3).

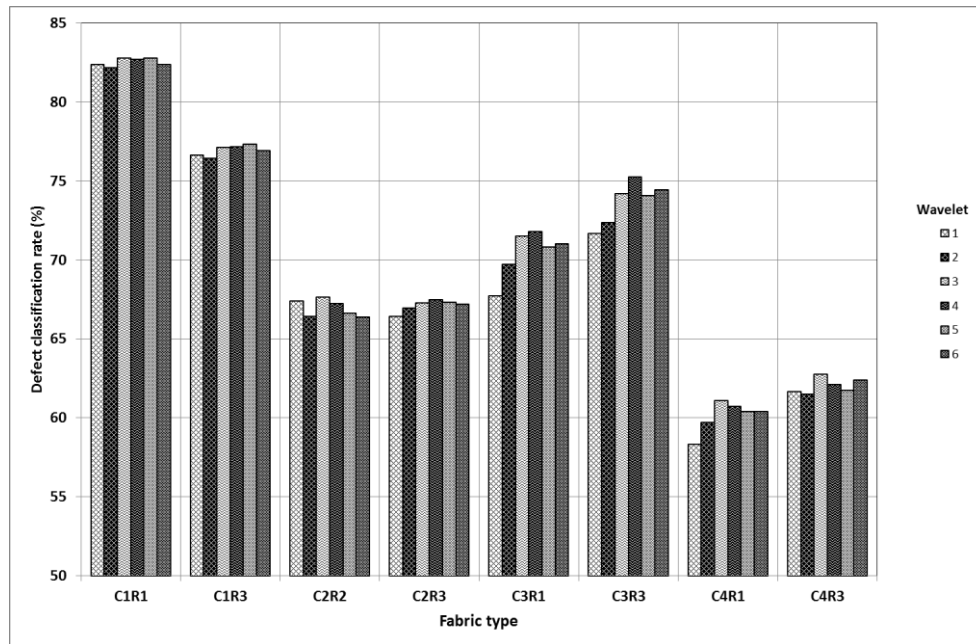


Figure 6-25: Detection classification rate of DTCWT-based features as the wavelet filter length increases for the different types of fabric when a K-nearest-neighbour classifier is used

Table 6-20: Average defect classification rate (%) of DTCWT-based features as the wavelet filter length increases for the eight types of fabrics when a K-nearest-neighbour classifier is used

Fabric type	Wavelet	1	2	3	4	5	6
	Filter length	4	6	8	10	12	14
C1R1		82.4	82.2	82.8	82.7	82.8	82.4
C1R3		76.6	76.4	77.1	77.2	77.3	76.9
C2R2		67.4	66.5	67.7	67.3	66.7	66.4
C2R3		66.4	67.0	67.3	67.5	67.3	67.2
C3R1		67.8	69.7	71.5	71.8	70.8	71.0
C3R3		71.7	72.4	74.2	75.3	74.1	74.5
C4R1		58.3	59.7	61.1	60.7	60.4	60.4
C4R3		61.7	61.5	62.8	62.1	61.8	62.4

Comparing the defect classification performance of the three different classifiers used in this thesis, the K-nearest-neighbour classifier and the feed-forward neural network classifiers have

classification rates in the same range and higher than those obtained using the MCE-trained Euclidean distance classifier.

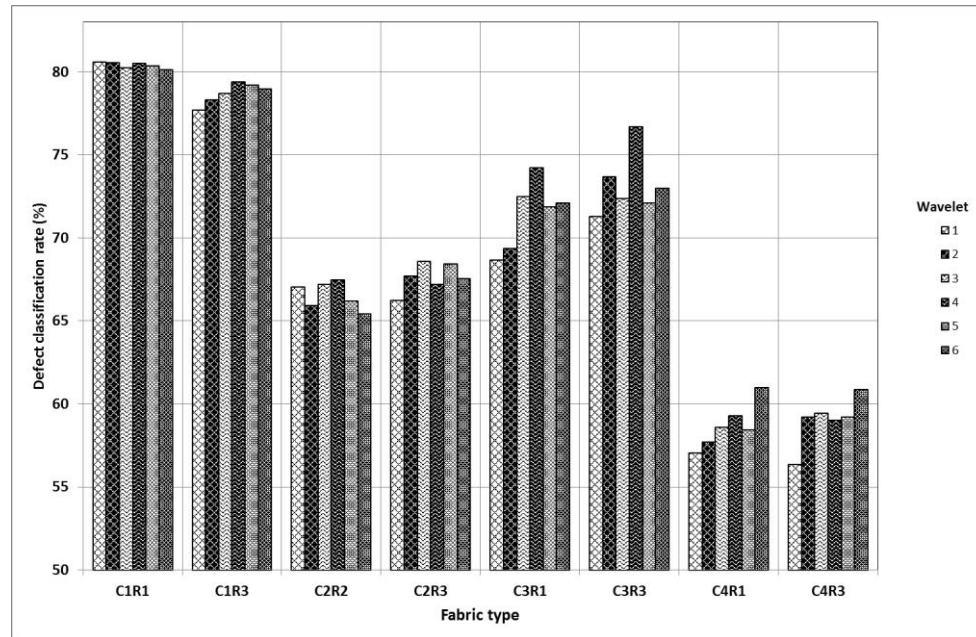


Figure 6-26: Detection classification rate of DTCWT-based features as the wavelet filter length increases for the different types of fabric when a feed-forward neural network classifier is used

Table 6-21: Average defect classification rate (%) of DTCWT-based features as the wavelet filter length increases for the eight types of fabrics when a feed-forward neural network classifier is used

Fabric type	Wavelet	1	2	3	4	5	6
	Filter length	4	6	8	10	12	14
C1R1		80.6	80.5	80.2	80.5	80.3	80.1
C1R3		77.7	78.3	78.7	79.4	79.2	79.0
C2R2		67.0	65.9	67.2	67.5	66.2	65.4
C2R3		66.2	67.7	68.6	67.2	68.4	67.6
C3R1		68.7	69.3	72.5	74.2	71.9	72.1
C3R3		71.3	73.7	72.4	76.7	72.1	73.0
C4R1		57.0	57.7	58.6	59.3	58.5	61.0
C4R3		56.4	59.2	59.5	59.0	59.2	60.9

6.12.3 Summary of best wavelet-based features for defect classification

Table 6-22 summarises the information on the best wavelet-based features for defect classification when three different classifiers are used. DTCWT-based features are best for most of the fabric types. The exception is printed fabrics, with no apparent periodicity for

which UDWT-based features generally perform best. The same observation was made for the defect detection results in Section 6.7.

Table 6-22: Comparison of best wavelet-based defect classification performance of three different classifiers

Fabric type	MCE-trained Euclidean distance classifier			K-nearest-neighbour classifier			Feed-forward neural network classifier		
	Transform type	Wavelet	Best class. rate (%)	Transform type	Wavelet	Best class. rate (%)	Transform type	Wavelet	Best class. rate (%)
C1R1	DTCWT	4	74.7	DTCWT	3	82.8	DTCWT	1	80.6
C1R3	DTCWT	1	66.9	DTCWT	5	77.3	DTCWT	4	79.4
C2R2	DTCWT	4	59.6	DTCWT	3	67.7	DTCWT	4	67.5
C2R3	DTCWT	3	49.4	DTCWT	4	67.5	DTCWT	3	68.6
C3R1	DTCWT	4	55.7	DTCWT	4	71.8	DTCWT	4	74.2
C3R3	DTCWT	5	53.6	DTCWT	4	75.3	DTCWT	4	76.7
C4R1	UDWT	5	54.4	UDWT	4	63.9	UDWT	2	69.4
C4R3	UDWT	3	53.2	DTCWT	3	62.8	UDWT	2	66.6

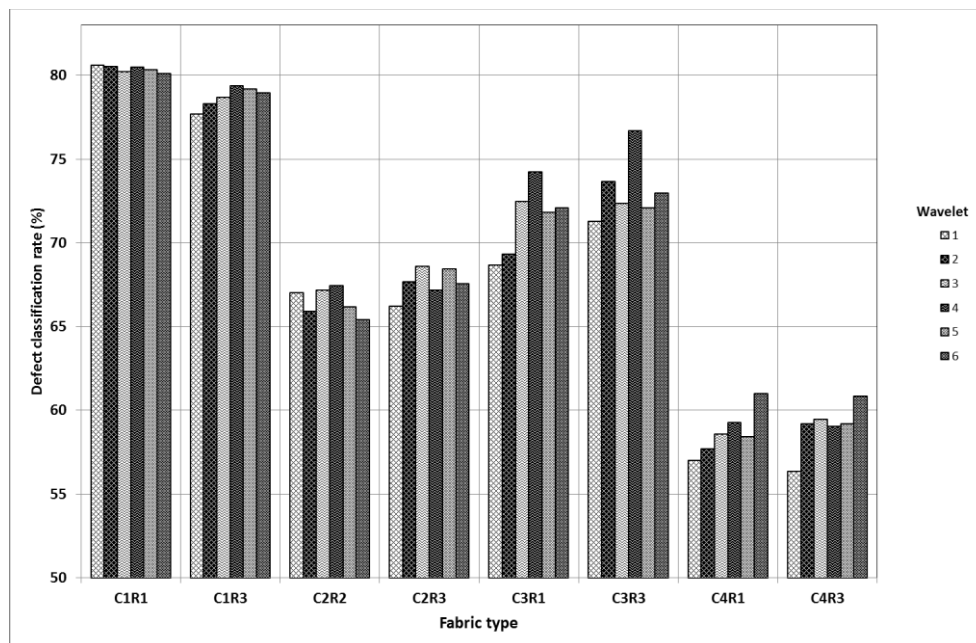


Figure 6-27: Detection classification rate of DTCWT-based features as the wavelet filter length increases for the different types of fabric when feed-forward neural network classifier is used

Table 6-23: Average defect classification rate (%) of DTCWT-based features as the wavelet filter length increases for the eight types of fabrics when a feed-forward neural network classifier is used

Fabric type	Wavelet					
	1	2	3	4	5	6
C1R1	80.6	80.5	80.2	80.5	80.3	80.1
C1R3	77.7	78.3	78.7	79.4	79.2	79.0
C2R2	67.0	65.9	67.2	67.5	66.2	65.4
C2R3	66.2	67.7	68.6	67.2	68.4	67.6
C3R1	68.7	69.3	72.5	74.2	71.9	72.1
C3R3	71.3	73.7	72.4	76.7	72.1	73.0
C4R1	57.0	57.7	58.6	59.3	58.5	61.0
C4R3	56.4	59.2	59.5	59.0	59.2	60.9

6.13 Summary

In this chapter we studied the performance of wavelet-based features for fabric defect detection and classification. Due to the fact that pattern recognition applications of wavelet transforms such as fabric defect detection and classification require the use of shift-invariant wavelet transforms, we chose both the undecimated discrete wavelet transform (UDWT) and the dual-tree complex wavelet transform (DTCWT), the two best known solutions for the problem of shift-variance of the wavelet transform.

We first designed the filters required for implementation of both UDWT and DTCWT. As we wanted to study the effect of filter length (or wavelet size of support) on the defect detection performance of the derived features, the design was done for different filter lengths from 2 through 16 in increments of two for UDWT and from 4 through 14 in increments of two for the DTCWT.

Experimental defect detection results showed that there is a clearly visible decreasing trend in the defect detection rate as the filter length (or the wavelet size of support) increases for UDWT. For DTCWT the variation of the defect detection rate with respect to the length of the filters was less evident, although a slight increasing trend could be noticed. However, these general observations, made from the averaged defect detection rates across different types of fabrics, were not uniform for individual fabric types.

The comparison of defect detection performance of UDWT- and DTCWT-based features showed that DTCWT-based ones outperformed those based on UDWT for most types of fabrics, with the exception of the printed fabrics with no apparent periodicity (class C4) where

UDWT-based features performed better. A possible explanation is the low regularity of those types of fabrics in different directions as explained in Section 6.7.

Regarding the number of levels of wavelet decomposition, experimental results showed that all five levels of decomposition were useful for DTCWT-based features, while three levels were enough for UDWT-based features. The two additional levels of UDWT decomposition would be too computationally expensive for the small increment in the resulting detection rates.

All the above observations were made based on experimental results obtained using the MCE-trained Euclidean distance classifier. To verify their validity with different classifiers, we performed the same experiments, but using (i) the K-nearest-neighbour classifier (with $K=3$) and (ii) the feed-forward neural network classifier. The results revealed that those observations remained valid. This suggests that they could be independent of classifiers, but more experiments with more classifiers are required to confirm this possibility.

When used for defect classification the same decreasing trend of the UDWT-based features with respect to the wavelet filter length can be observed. However, this trend is less consistent from one classifier type to another. DTCWT-based features generally show little variation in defect classification performance with respect to the wavelet filter length, apart from for fabric types C3R1, C3R3, C4R1 and C4R3, where an increasing trend is observed. Similar to the defect detection results, the DTCWT-based features outperform the UDWT-based features in the defect classification results for most fabric types apart from the printed fabrics with no apparent periodicity (Class C4).

Chapter 7: Exploration of Markov random field-based features

7.1 Introduction

Model-based texture analysing methods such as Markov random fields (MRF) attempt to model texture as a stochastic process. This approach has the advantage that it is capable of synthesising textures that match the observed textures from models [20]. This contrasts with methods such as grey level co-occurrence matrix (GLCM) and wavelets, which are essentially feature-based with no intention of producing texture from the model. This suggests that the resulting features have better texture discriminating ability.

In this chapter we investigate the performance of MRF features for fabric defect detection and classification. The objective is to select MRF features that can be combined with GLCM and wavelet features for improved fabric defect detection and classification.

This chapter will be organised as follows. In Section 7.2 we decide which MRF model to use. In Section 7.3 we present the MRF feature extraction algorithm. Section 7.4 studies the defect detection performance of MRF features as function of the model order. Section 7.6 deals with the problem of fabric defect classification using MRF features, while Section 7.7 summarises the chapter.

7.2 Which Markov random field model?

As described in Subsection 3.4.3, many different types of Markov random field models can be set up by choosing a form of the energy function $U(f)$ given by (3.97). However, the auto-models are the most frequently used in practice.

In this chapter, we will use the Gaussian Markov random field (GMRF) model because it is the most widely used for the modelling of various man-made and natural textures [184]. Moreover, its effectiveness has been shown in various publications [20, 111]. We will use the model to study its effectiveness in detecting defects in fabric images as the order of the model increases from 1 through 9. We will also use it for defect classification and the observations we obtain from the experiments will allow us to choose the features that will be combined with GLCM and wavelet features for improved effectiveness in defect detection and classification.

7.3 MRF feature extraction algorithm

As described in Subsection 3.4.5, MRF feature extraction of a texture image is equivalent to parameter estimation of the MRF model of that image.

Figure 7-1 shows the neighbourhood of a pixel X. In the number identifying each pixel of the neighbourhood, the first digit refers to the order of the model, while the second digit is the serial number of the pair of symmetric pixels with respect to X.

91	82	72	62	73	83	92
81	51	42	32	43	52	84
71	41	21	12	22	44	74
61	31	11	X	11	31	61
74	44	22	12	21	41	71
84	52	43	32	42	51	81
92	83	73	62	72	82	91

Figure 7-1: Neighbourhood of point X

As described in Subsection 3.4.3, a GMRF model may be represented as a non-causal autoregressive model given by (3.108) and (3.109). Let \bar{f} be the image obtained after subtracting from the original image f its mean value μ as given by (7.1).

$$\bar{f} = f - \mu \quad (7.1)$$

Then (3.108) can be rewritten as (7.2) for the first order model

$$\bar{f}_{m,n} = \beta_{11}(\bar{f}_{m,n-1} + \bar{f}_{m,n+1}) + \beta_{12}(\bar{f}_{m-1,n} + \bar{f}_{m+1,n}) + e_{m,n} \quad (7.2)$$

where (m, n) are coordinates of the central pixel and $e_{m,n}$ is the value of the Gaussian noise at pixel (m, n) . β_{11} and β_{12} are the model parameters.

The second order model (3.108) can be rewritten as (7.3).

$$\begin{aligned} \bar{f}_{m,n} = & \beta_{11}(\bar{f}_{m,n-1} + \bar{f}_{m,n+1}) + \beta_{12}(\bar{f}_{m-1,n} + \bar{f}_{m+1,n}) + \\ & \beta_{21}(\bar{f}_{m-1,n-1} + \bar{f}_{m+1,n+1}) + \beta_{22}(\bar{f}_{m+1,n-1} + \bar{f}_{m-1,n+1}) + e_{m,n} \end{aligned} \quad (7.3)$$

Equations similar to (7.2) and (7.3) can be easily written for higher order models.

To estimate the model parameters for each model order from 1 through 9 we used the algorithm shown in Table 7-2 in form of pseudo code.

In step 5) of the pseudo code, the number of parameters $\beta_{k,l}$ depends on the order of the GMRF model as shown in Table 7-1.

Table 7-1: Number of GMRF model parameters

Order of the GMRF model	1	2	3	4	5	6	7	8	9
Number of parameters $\beta_{k,l}$	2	4	6	10	12	14	18	22	24

In step 7) solving the system of overdetermined linear equations by LSE is equivalent to linear regression [185] and therefore the MATLAB function '*regress*' was used to estimate the parameters $\beta_{k,l}$ as well as the residual error $e_{m,n}$.

The calculation of the predicted value of a pixel $\hat{f}(m,n)$ given its neighbourhood (step 9) is performed using (7.4) for first order models and (7.5) for second order models. Similar equations are used for higher order models.

$$\hat{f}_{m,n} = \hat{\beta}_{11}(\bar{f}_{m,n-1} + \bar{f}_{m,n+1}) + \hat{\beta}_{12}(\bar{f}_{m-1,n} + \bar{f}_{m+1,n}) \quad (7.4)$$

$$\begin{aligned} \hat{f}_{m,n} = & \hat{\beta}_{11}(\bar{f}_{m,n-1} + \bar{f}_{m,n+1}) + \hat{\beta}_{12}(\bar{f}_{m-1,n} + \bar{f}_{m+1,n}) + \\ & \hat{\beta}_{21}(\bar{f}_{m-1,n-1} + \bar{f}_{m+1,n+1}) + \hat{\beta}_{22}(\bar{f}_{m+1,n-1} + \bar{f}_{m-1,n+1}) \end{aligned} \quad (7.5)$$

Where $\hat{\beta}_{k,l}$ are the estimates of the model parameters $\beta_{k,l}$ found in step 7).

Table 7-2: Algorithm for GMRF feature extraction from a given image f and for a given model order

- 1) Calculate the mean grey scale value μ of the image f .
- 2) Calculate \bar{f} by using (7.1).
- 3) Symmetrically extend image \bar{f} so that the central pixel in the neighbourhood in Figure 7-1 (for the given order) can occupy all the positions of the original image and keep all the neighbourhood pixels within the image.
- 4) For each pixel $\bar{f}_{m,n}$ consider its neighbourhood as shown in Figure 7-1, where $\bar{f}_{m,n}$ is the central pixel.
- 5) Write equations similar to (7.2) and (7.3) for the given model order. $\beta_{k,l}$ is the unknown to be estimated, $\bar{f}_{m,n}$ is value of the current central pixel, and the coefficients of the unknown $\beta_{k,l}$ are the sums of corresponding pixel values which are symmetrically opposed with respect to the central pixel. In Figure 7-1 such pixels are similarly numbered. The Gaussian noise is not considered in this step.
- 6) Continue with the next pixel.
- 7) After completing steps 4), 5) and 6) for all the pixels of \bar{f} , the obtained equations form an overdetermined system of linear equations where the model parameters are the unknowns $\beta_{k,l}$ to be determined. Solve the equations using the LSE method.
- 8) For each pixel position (m, n)
- 9) Calculate the model estimate of central pixel $\hat{f}(m, n)$ using equation similar to (7.4) or (7.6) for the given model.
- 10) Calculate the residual error $e_{m,n}$ as $e_{m,n} = \bar{f}_{m,n} - \hat{f}_{m,n}$
- 11) Continue with the next pixel position.
- 12) After completing the steps 8) through 11) for all the pixel positions in f , calculate the variance of the residual errors $\sigma^2 = var(e_{m,n})$
- 13) Consider the set $\{\mu, \sigma^2, \hat{\beta}_{k,l}\}$ as the GMRF features of the texture image f .

7.4 Defect detection performance as function of the model order

Defect detection experiments were performed using features derived from the MRF models of orders from 1 through 9 for all eight fabric types in our dataset. The MCE-trained Euclidean distance classifier was used.

The results are shown in Figure 7-2 and Table 7-3. We can read from the results that the first order MRF is by far the most effective model for detecting defect in fine fabrics (C1R1, C1R3, C2R2 and C2R3) compared to the models of the other orders. For these types of fabrics the defect detection performance tends to decrease as the order of the model increases.

For fabrics with a periodic structure (C3R1 and C3R3) the average defect detection rate remains low (below 67%) and does not change much from one model order to the next. The best average detection rate is 66.6% for fabric type C3R1, recorded when the first order model is used, and 64.6% for fabric type C3R3 recorded when the seventh order model is used.

For printed fabrics with no apparent periodicity (C4R1 and C4R3) the defect detection performance tends to slowly increase with the model order, even though the second order seems to be an exception for fabric type C4R1 and the third order seems to be an exception for fabric type C4R3. The best average defect detection rate of 67.5% is observed for fabric type C4R1 when the model order is 2, and 71.1% observed for fabric C4R3 when the model order is 9. However, in the latter case we see that the features obtained using the MRF model of order 3 performs almost equally well (70.7%). Therefore, in combination experiments done in chapter 8, we will use the features obtained using the third model because they are much faster to compute than those from the ninth order model.

Table 7-4 summarises the best MRF features for the different fabric types.

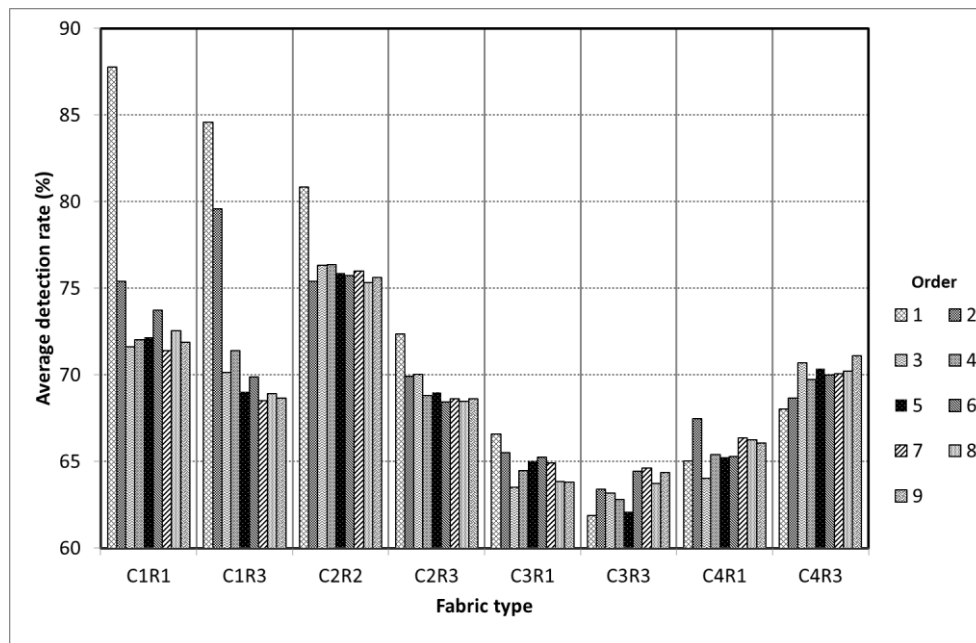


Figure 7-2: Defect detection performance of MRF features as the order of the model changes for different fabric types.

Table 7-3: Average defect detection rate (%) for different fabric as the order of the MRF model used to find texture features varies from 1 through 9

Fabric type	Order of the MRF model								
	1	2	3	4	5	6	7	8	9
C1R1	87.8	75.4	71.6	72.0	72.2	73.8	71.4	72.5	71.9
C1R3	84.6	79.6	70.1	71.4	69.0	69.9	68.5	68.9	68.7
C2R2	80.8	75.4	76.3	76.4	75.9	75.7	76.0	75.3	75.6
C2R3	72.4	69.9	70.0	68.8	69.0	68.5	68.6	68.5	68.6
C3R1	66.6	65.5	63.5	64.5	65.0	65.3	64.9	63.8	63.8
C3R3	61.9	63.4	63.2	62.8	62.1	64.5	64.6	63.7	64.4
C4R1	65.0	67.5	64.0	65.4	65.3	65.3	66.4	66.2	66.1
C4R3	68.0	68.6	70.7	69.7	70.4	70.0	70.1	70.2	71.1

Table 7-4: Best MRF-based features per fabric type

Fabric type	MRF model order	Best average detection rate (%)
C1R1	1	87.8
C1R3	1	84.6
C2R2	1	80.8
C2R3	1	72.4
C3R1	1	66.6
C3R3	7	64.6
C4R1	2	65.5
C4R3	3	70.7

7.5 Effect of the classifier on the performance of MRF features

The observations made in Section 7.4 about the dependence of the MRF features on the order of the model were based on the use of the MCE-trained Euclidean distance classifier. We specifically identified the best model order to use for each of the eight fabric types in our dataset. In this section we investigate whether or not the same observations hold when a K-nearest-neighbour classifier (K=3) and a feed-forward neural network classifier are used. We also compare the best defect detection performance obtained when each of the three classifiers are used. For the neural network classifier the number of inputs is equal to the number of MRF-based features and varies from 4 to 26 depending on the order of the model. For the number of neurons in the hidden layer, we used the formula given by (5.4). We used the training parameters and procedure for the neural network classifier as explained in Subsection 5.6.2.

Figure 7-3 and Table 7-5 show how the average defect detection rate varies as the order of the GMRF model increases when a K-nearest-neighbour classifier is used, while Figure 7-4 and Table 7-6 show the same information when a feed-forward neural network classifier is used. We can see from both sets of results that, for fine fabrics (C1R1, C1R3, C2R2 and C2R3), the same trend of decreasing detection performance as the order of the model used is increased is still maintained. That observation was made in Section 7.4 when a MCE-trained classifier is used. The first order is also still the best model for those types of fabrics, with the exception of the fabric type C2R3 for which the best detection performance is observed for the fourth model when a feed-forward neural network classifier is used. The overwhelming performance of the first order with respect to the other models is observed only for the K-nearest-neighbour classifier and for only the very fine fabrics (C1R1 and C1R3), not for the feed-forward neural network classifier.

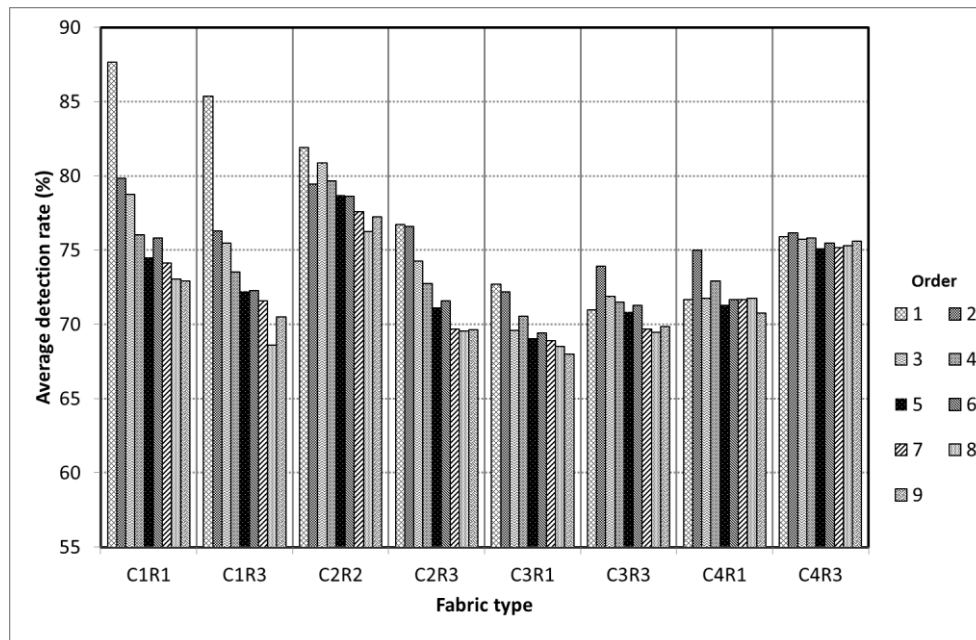


Figure 7-3: Defect detection performance of MRF features as the order of the model changes for different fabric types when a K-nearest-neighbour classifier is used

Table 7-5: Average defect detection rate (%) for different fabric as the order of the MRF model used to find texture features varies from 1 through 9 and when a K-nearest-neighbour classifier is used

Fabric type	Order of the MRF model								
	1	2	3	4	5	6	7	8	9
C1R1	87.7	79.9	78.8	76.0	74.5	75.8	74.1	73.1	72.9
C1R3	85.4	76.3	75.5	73.5	72.2	72.3	71.6	68.6	70.5
C2R2	81.9	79.5	80.9	79.7	78.7	78.6	77.6	76.3	77.3
C2R3	76.7	76.6	74.3	72.8	71.1	71.6	69.7	69.6	69.6
C3R1	72.7	72.2	69.6	70.5	69.1	69.4	68.9	68.5	68.0
C3R3	71.0	73.9	71.9	71.5	70.9	71.3	69.7	69.5	69.9
C4R1	71.7	75.0	71.8	72.9	71.3	71.7	71.7	71.7	70.8
C4R3	75.9	76.2	75.8	75.8	75.1	75.5	75.2	75.3	75.6

For fabrics with a periodic structure (C3R1 and C3R3), low-order models (first and second order) perform better when the K-nearest-neighbour classifier is used, while the dependence of the performance on the model order is less regular when a feed-forward neural network classifier is used.

Finally, for fabrics with no apparent periodicity (C4R1 and C4R3), we observe the same increasing trend of the defect detection rate as the order of the model increases when the feed-forward neural network is used, but not when the K-nearest-neighbour classifier is used. That indicates greater ability of neural networks in conjunction with high-order MRF models to capture statistical relationship of pixels of bigger texture primitives present in fabric types of class 4. In the latter case of the K-nearest-neighbour classifier, we observe a relatively high performance for the second order model for fabric type C4R1, and an almost constant performance for all nine model orders for fabric type C4R3.

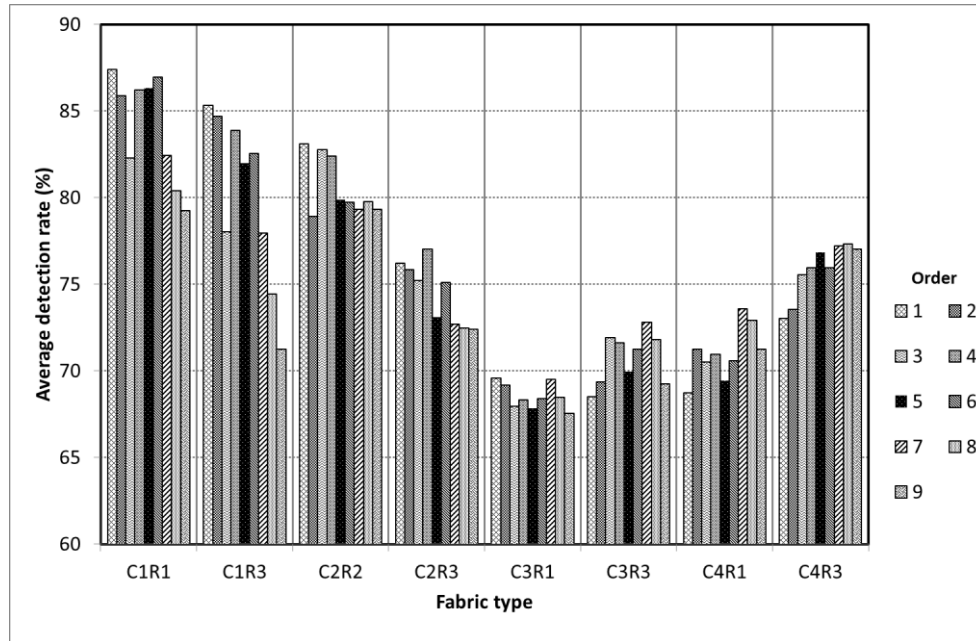


Figure 7-4: Defect detection performance of MRF features as the order of the model changes for different fabric types when a feed-forward neural network classifier is used

Table 7-6: Average defect detection rate (%) for different fabric as the order of the MRF model used to find texture features varies from 1 through 9 and when a feed-forward neural network is used

Fabric type	Order of the MRF model								
	1	2	3	4	5	6	7	8	9
C1R1	87.4	85.9	82.3	86.2	86.3	86.9	82.4	80.4	79.2
C1R3	85.3	84.7	78.0	83.9	82.0	82.6	77.9	74.4	71.3
C2R2	83.1	78.9	82.8	82.4	79.9	79.7	79.3	79.8	79.3
C2R3	76.2	75.8	75.2	77.0	73.1	75.1	72.7	72.5	72.4
C3R1	69.6	69.2	68.0	68.3	67.9	68.4	69.5	68.5	67.6
C3R3	68.5	69.4	71.9	71.6	70.0	71.3	72.8	71.8	69.2
C4R1	68.8	71.3	70.5	70.9	69.4	70.6	73.6	72.9	71.3
C4R3	73.0	73.5	75.6	76.0	76.8	76.0	77.2	77.3	77.0

Table 7-7 compares the defect detection performance of the MCE-trained Euclidean distance classifier, K-nearest-neighbour classifier and the feed-forward neural network classifiers fed with MRF-based features. In this comparison the best detection rate across different orders of the GMRF models is used. The three classifiers perform almost equally, apart from the MCE-trained Euclidean distance classifier which shows lower performance than the two other

classifiers for fabrics with a periodic structure and for fabrics with no apparent periodicity (C3R1, C3R3, C4R1 and C4R3).

Table 7-7: Comparison of best MRF-based defect detection performance of three different classifiers

Fabric type	MCE-trained Euclidean distance classifier		K-nearest-neighbour classifier		Feed-forward neural network classifier	
	MRF model order	Best average detection rate (%)	MRF model order	Best average detection rate (%)	MRF model order	Best average detection rate (%)
C1R1	1	87.8	1	87.7	1	87.4
C1R3	1	84.6	1	85.4	1	85.3
C2R2	1	80.8	1	81.9	1	83.1
C2R3	1	72.4	1	76.7	4	77.0
C3R1	1	66.6	1	72.7	1	69.6
C3R3	7	64.6	2	73.9	7	72.8
C4R1	2	65.5	2	75.0	7	73.6
C4R3	3	70.7	2	76.2	8	77.3

7.6 Fabric defect classification using MRF features

In this section we perform experiments in order to identify for each fabric type the GMRF model order that provides features able to best discriminate the four defect classes in addition to the defect-free class.

For each model order from 1 through 9, and for each the eight fabric types, the defect classification experiment illustrated in Figure 5-18 is performed. The experiment is done using (i) the MCE-trained Euclidean distance classifier, (ii) the K-nearest-neighbour classifier and (iii) the feed-forward neural network. The defect classification rates for each type of fabric are then recorded and analysed.

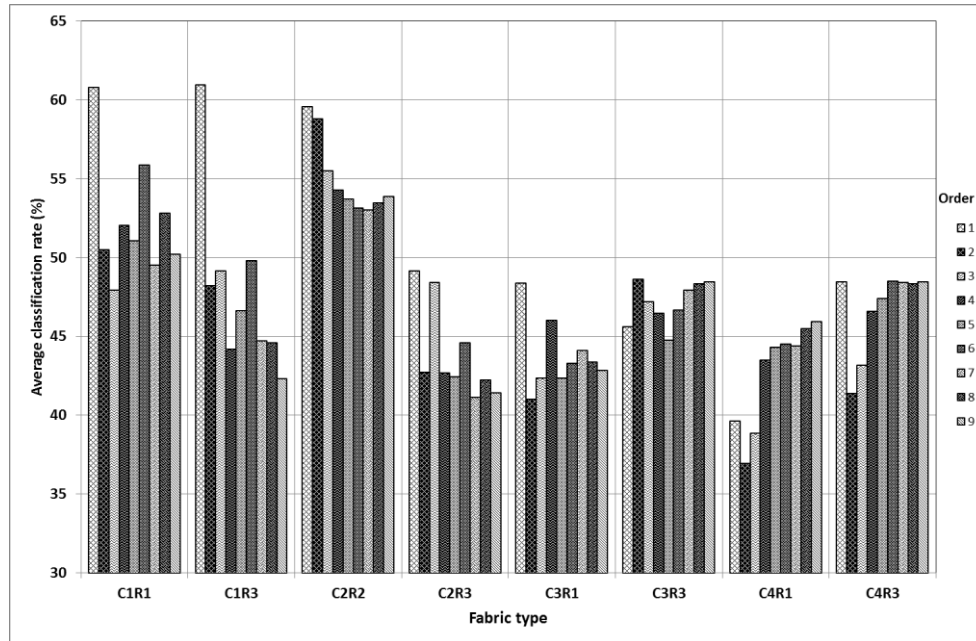


Figure 7-5: Defect classification performance of MRF features as the order of the model changes when a MCE-trained Euclidean distance classifier is used

Table 7-8: Average defect classification rate (%) for different fabric as the order of the MRF model changes when a MCE-trained Euclidean distance classifier is used

Fabric type	Order of the MRF model								
	1	2	3	4	5	6	7	8	9
C1R1	60.8	50.5	48.0	52.0	51.1	55.9	49.5	52.8	50.2
C1R3	61.0	48.2	49.2	44.2	46.7	49.8	44.7	44.6	42.3
C2R2	59.6	58.8	55.5	54.3	53.7	53.2	53.0	53.5	53.9
C2R3	49.2	42.7	48.4	42.7	42.5	44.6	41.2	42.2	41.4
C3R1	48.4	41.0	42.4	46.0	42.4	43.3	44.1	43.4	42.8
C3R3	45.6	48.6	47.2	46.5	44.8	46.7	47.9	48.3	48.5
C4R1	39.6	37.0	38.9	43.5	44.3	44.5	44.4	45.5	46.0
C4R3	48.5	41.4	43.2	46.6	47.4	48.5	48.4	48.4	48.5

The results from Figure 7-5 and Table 7-8 show that the classification rates are generally low; the highest is 61.0%, obtained for fabric type C1R3 using features from the first order model. That is normal as classification experiments involve five classes while detection experiments involve only two classes. The results also show that the first order model provides the best features for defect classification, and the only exceptions are fabric types C3R3 and C4R1 where the best features are provided by the second and the ninth model respectively. This generally agrees with the detection results shown previously in this chapter.

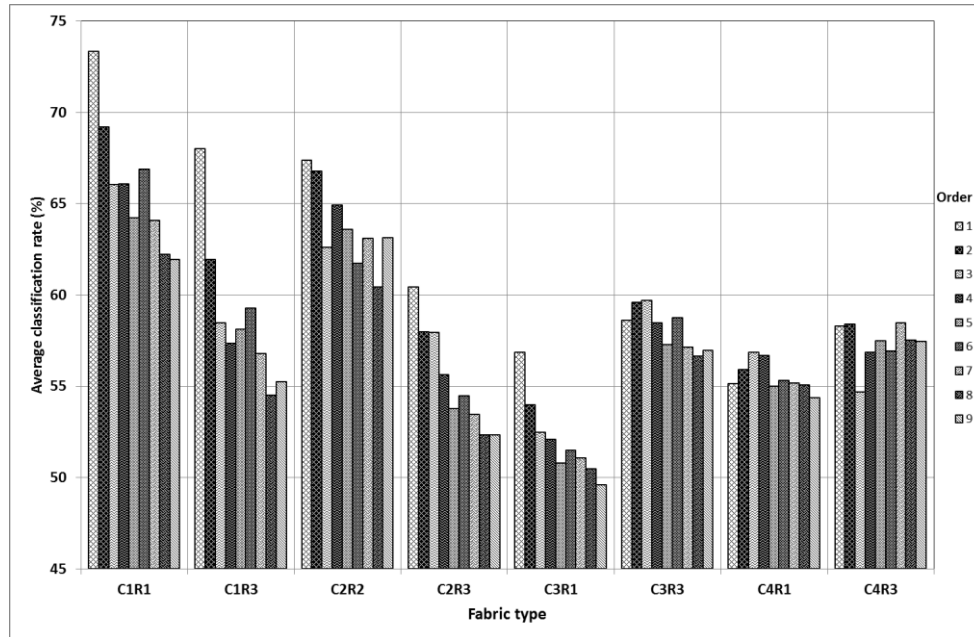


Figure 7-6: Defect classification performance of MRF features as the order of the model changes when a K-nearest-neighbour classifier is used

Table 7-9: Average defect classification rate (%) for different fabric as the order of the MRF model changes when a K-nearest-neighbour classifier is used

Fabric type	Order of the MRF model								
	1	2	3	4	5	6	7	8	9
C1R1	73.4	69.2	66.1	66.1	64.2	66.9	64.1	62.2	62.0
C1R3	68.0	61.9	58.5	57.4	58.1	59.3	56.8	54.5	55.3
C2R2	67.4	66.8	62.6	64.9	63.6	61.7	63.1	60.4	63.1
C2R3	60.4	58.0	58.0	55.6	53.8	54.5	53.5	52.3	52.4
C3R1	56.9	54.0	52.5	52.1	50.8	51.5	51.1	50.5	49.6
C3R3	58.6	59.6	59.7	58.5	57.3	58.8	57.1	56.7	57.0
C4R1	55.2	55.9	56.9	56.7	55.0	55.3	55.2	55.1	54.4
C4R3	58.3	58.4	54.7	56.9	57.5	56.9	58.5	57.5	57.5

Figure 7-6 and Table 7-9 show that the same trend is kept when a K-nearest-neighbour classifier is used as the features derived from the first order model perform the best for most fabric types. However, the detection rates are higher than those obtained using the MCE-trained Euclidean distance classifier.

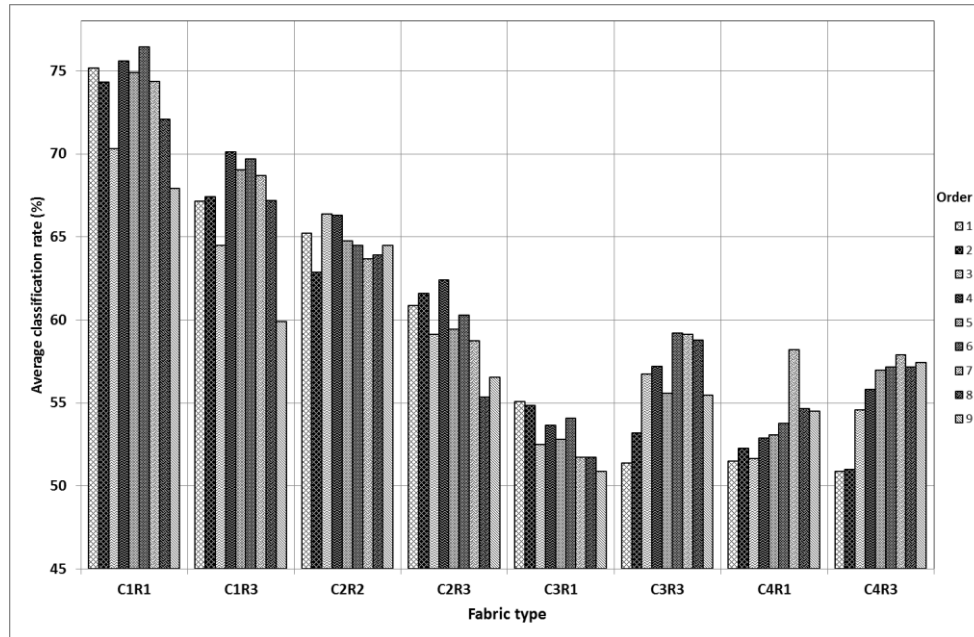


Figure 7-7: Defect classification performance of MRF features as the order of the model changes when a feed-forward neural network classifier is used

Table 7-10: Average defect classification rate (%) for different fabric as the order of the MRF model changes when a feed-forward neural network classifier is used

Fabric type	Order of the MRF model								
	1	2	3	4	5	6	7	8	9
C1R1	75.2	74.3	70.3	75.6	74.9	76.5	74.4	72.1	67.9
C1R3	67.2	67.4	64.5	70.1	69.0	69.7	68.7	67.2	59.9
C2R2	65.2	62.9	66.4	66.3	64.8	64.5	63.7	63.9	64.5
C2R3	60.9	61.6	59.1	62.4	59.5	60.3	58.8	55.4	56.6
C3R1	55.1	54.8	52.5	53.7	52.8	54.1	51.7	51.7	50.9
C3R3	51.4	53.2	56.7	57.2	55.6	59.2	59.1	58.8	55.5
C4R1	51.5	52.3	51.7	52.9	53.1	53.8	58.2	54.7	54.5
C4R3	50.9	51.0	54.6	55.8	57.0	57.2	57.9	57.2	57.5

When we look at the defect classification rates obtained using the feed-forward neural network shown in Figure 7-7 and Table 7-10, we notice that the defect classification rates are generally in the same range as those obtained using the K-nearest-neighbour classifier, but the best performing features are no longer the same for the different types of fabrics. Features derived from model orders between 3 and 7 perform the best. These results suggest that the best performing features for defect classification may depend not only on the features themselves but also on the classifier used.

Table 7-11 summarises the best performing features when the three different classifiers are used.

Table 7-11: Comparison of best MRF-based defect classification performance of three different classifiers

Fabric type	MCE-trained Euclidean distance classifier		K-nearest-neighbour classifier		Feed-forward neural network classifier	
	MRF model order	Best average classification rate (%)	MRF model order	Best average classification rate (%)	MRF model order	Best average classification rate (%)
C1R1	1	60.8	1	73.4	6	76.5
C1R3	1	61.0	1	68.0	4	70.1
C2R2	1	59.6	1	67.4	3	66.4
C2R3	1	49.2	1	60.4	4	62.4
C3R1	1	48.4	1	56.9	1	55.1
C3R3	2	48.6	3	59.7	6	59.2
C4R1	9	46.0	3	56.9	7	58.2
C4R3	6	48.5	7	58.5	7	57.9

7.7 Summary

In this chapter we studied the defect detection and classification performance of features derived from the Markov random field models. We chose the Gaussian Markov random field models and studied the variation of the defect detection rate as the model order varied from 1 through 9 for the different types of fabrics in our dataset.

The results obtained using three different classifiers, (i) the MCE-trained Euclidean distance classifier, (ii) the K-nearest-neighbour classifier ($K=3$) and (iii) the feed-forward neural network classifier, revealed that features from the low-order models (1, 2, 3) – especially the first order – performed best for most of the fabric types. The exceptions were fabric type C3R3, where the defect detection rate of the seventh order model provided the best features when the MCE-trained classifier was used, and C4R1 and C4R3 (printed fabric with no apparent periodicity) for which respectively the seventh and the eighth order models scored the highest. That may be due to the fact that high order models involve distant pixels that might be more statistically related in fabric types C3R3, C4R1 and C4R3 than in other fabric types.

The defect classification results using MRF-based features show the same trend of low-order models providing the best features for most of the fabrics when the MCE-trained Euclidean distance and the K-nearest-neighbour classifiers are used. However, when the feed-forward

neural network classifier is used, the features from model orders between 3 and 7 perform best.

Chapter 8: Combination schemes and their performance

8.1 Introduction

As stated earlier the objective of this thesis is to propose fabric defect detection and classification methods by combining the GLCM, wavelet and MRF-based methods. In chapters 5, 6 and 7 we discussed the performance of fabric defect detection and classification of each of the three methods individually and identified the most effective features for each of the fabric types in our experimental dataset.

In this chapter we propose three combination schemes of these methods that lead to more accurate defect detection and classification than each of the methods individually. In the first combination scheme, which we will refer to as CS1, we pool together the GLCM, wavelet-based and MRF-based best features for each type of fabric and perform some dimensionality reduction on them before feeding them into a classifier.

In the second combination scheme, referred to as CS2, the fabric images under study are submitted to a three-level DTCWT-wavelet decomposition, then the best identified GLCM and MRF features are extracted from the wavelet sub-images. After dimensionality reduction performed separately on GLCM and MRF features, the resulting features are fed into a classifier.

The third combination scheme, referred to as CS3, is similar to CS2 with the exception that the GLCM and MRF features are not extracted from the wavelet coefficients themselves but from the reconstructed images from the wavelet coefficients.

In all the experiments in this chapter we started with the MCE-trained Euclidean distance classifier and then performed the same experiments using the K-nearest-neighbour classifier and feed-forward neural classifier. That is in line with the procedural order of experiments performed on individual features in previous chapters where the MCE-trained Euclidean distance classifier was used to make primary observations on their performance for defect detection and classification and then the K-nearest-neighbour and the feed-forward neural network classifiers were used to check the validity of the observations made as the classifier changes.

This chapter is organised as follows. In sections 8.2 through 8.4 we study the combination schemes CS1, CS2 and CS3 for fabric defect detection using the MCE-trained Euclidean distance classifier. In Section 8.5 we compare the defect detection performance of the three combination schemes using the MCE-trained Euclidean distance classifier for the different

types of fabrics in our dataset. Section 8.6 extends the study to the use of the K-nearest-neighbour and the feed-forward neural network classifiers. Section 8.7 deals with the problem of fabric defect classification using the combined schemes, while Section 8.8 summarises the chapter.

8.2 Direct combination of the best selected features (CS1)

8.2.1 Introduction

For the MCE-trained Euclidean distance classifier, we concluded in Section 5.6 that the optimal number of GLCM features for fabric defect detection is five, and we identified the five features for each of the eight fabric types in our dataset. In Section 6.10 we identified for each fabric types the most effective wavelet-based features in terms of type of wavelet transform (DTCWT or UDWT), wavelet size of support (among the six we proposed) and feature set (among two). Finally, in Section 7.4 we identified for each of the eight fabric types the most effective MRF features in terms of the MRF order model among nine. Table 8-1 summarises the results.

Table 8-1: Most effective features for the fabric defect detection methods to be combined when a MCE-trained Euclidean distance classifier is used

Fabric type	GLCM	Wavelet				MRF
	Optimal quintuplets of feature set	Wavelet transform type	Wavelet filter length	Number of levels	Feature set	Model order
C1R1	{f6, f10, f11, f13, f14}	DTCWT	4	5	Set c2	1
C1R3	{f2, f5, f7, f13, f14}	DTCWT	4	5	Set c2	1
C2R2	{f5, f8, f9, f11, f13}	DTCWT	4	5	Set c2	1
C2R3	{f2, f4, f11, f12, f14}	DTCWT	4	5	Set c1	1
C3R1	{f2, f6, f9, f11, f14}	DTCWT	8	5	Set c2	1
C3R3	{f2, f5, f9, f10, f14}	DTCWT	8	5	Set c1	7
C4R1	{f5, f10, f11, f13, f14}	UDWT	8	3	Set r2	2
C4R3	{f2, f3, f7, f8, f10}	UDWT	8	3	Set r2	3

Experiments in this section directly combine the features shown in Table 8-1 in order to improve defect detection accuracy. The number of GLCM features is ten because each feature in Table 8-1 represents a pair of sub-features (the mean and the range). The number of wavelet features is either 54 when the DTCWT is used or 30 if the UDWT is used. The number of MRF features depends on the order of the model, four for order 1, six for order 2, eight for order 3 and twenty for order 7. Due to the large number of wavelet-based features, we performed principal component analysis (PCA) [186] on them to reduce their dimensionality before combining them with GLCM and MRF features.

8.2.2 Experimental steps

The block diagram in Figure 8-1 shows how the GLCM-, wavelet- and MRF-based fabric defect detection methods are combined according to the first combination scheme. Blocks I and II were implemented as explained in chapter 5. Blocks III and IV were implemented as explained in chapter 6, while block VI was implemented as explained in chapter 7. Block VII regards the MCE-trained Euclidean distance classifier explained in Subsection 3.5.3.

The PCA in block V is a linear transformation that makes it possible to convert potentially correlated variables into a set of values (or new features) of linearly uncorrelated variables called principal components. The principal components are ordered so that the first few retain most of the variation present in the original features [186]. Therefore the classification accuracy should increase rapidly with the number of the retained principal components and then stabilise.

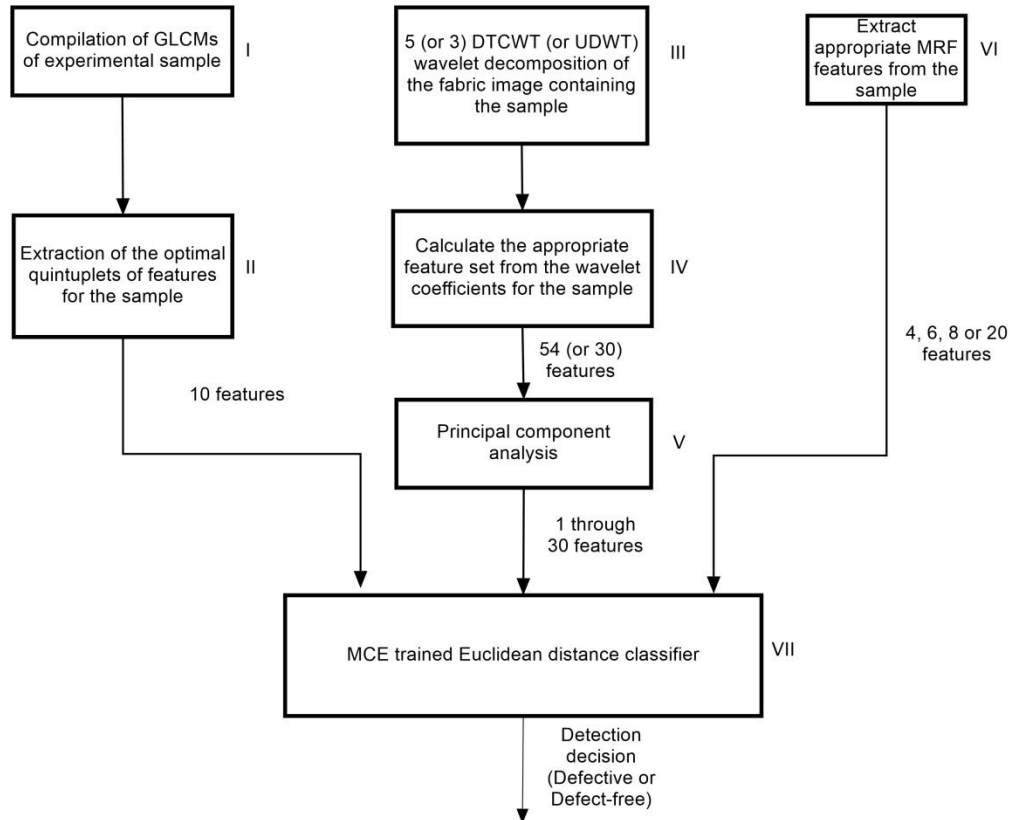


Figure 8-1: Block diagram of the first combination scheme (CS1)

The PCA was implemented using the MATLAB function '*processpca*'. In the training stage it takes the training features in matrix form and returns all the principal components of those features, as well as the settings used to perform the transform. These settings are then used by

the same function (*'processpca'*) with the testing features to return all the principal components of the testing features.

8.2.3 Results and interpretation

Figure 8-2 and shows the average detection rate obtained using the combined method CS1 as the number of principal components of wavelet features is increased from 0 through 30. We can see that the number of principal components to retain varies from one fabric type to the other. However, we can see that 30 principal components give the highest detection rate for all the fabric types in our dataset. Therefore, we decided to use 30 principal components of the wavelet-based features regardless of the fabric type.

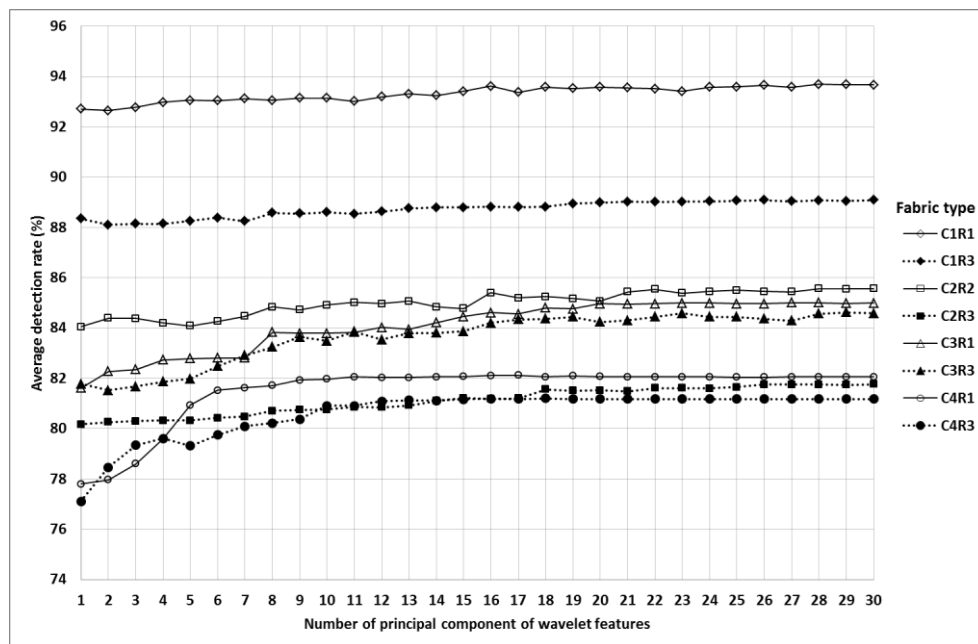


Figure 8-2: Defect detection performance of CS1 features for different fabric types as number of principal components of wavelet features changes

Table 8-2: Average defect detection rate (%) of CS1 features for different fabric types as number of principal components of wavelet features changes

Number of principal components of wavelet features	Fabric types							
	C1R1	C1R3	C2R2	C2R3	C3R1	C3R3	C4R1	C4R3
1	92.7	88.3	84.0	80.2	81.6	81.8	77.8	77.1
5	93.1	88.3	84.1	80.3	82.8	82.0	80.9	79.3
10	93.1	88.6	84.9	80.8	83.8	83.5	82.0	80.9
15	93.4	88.8	84.8	81.2	84.5	83.9	82.1	81.2
20	93.6	89.0	85.1	81.5	85.0	84.2	82.1	81.2
25	93.6	89.1	85.5	81.6	85.0	84.4	82.1	81.2
30	93.7	89.1	85.6	81.8	85.0	84.6	82.1	81.2

Figure 8-3 and Table 8-3 compare the defect detection performance of individual methods based on GLCM, wavelet and MRF features to that of the combined method CS1. Concerning the individual methods, we observe that the GLCM features perform the best, followed closely by the wavelet features, while the MRF features lag behind. This is true for most of the fabric types in our dataset. The exception is observed for fabric type C4R1 and C4R3, for which wavelet features perform better than both GLCM and MRF features. Our observations are similar to those made by Clausi [187] where he compared co-occurrence, Gabor and MRF features for classification of SAR sea ice imagery.

More importantly, we observe that the combined method CS1 performs better than any of the individual methods and that for all the fabric types in our dataset. We can therefore conclude that combining the GLCM, wavelet and MRF features according to the combination scheme CS1 effectively improves the defect detection performance.

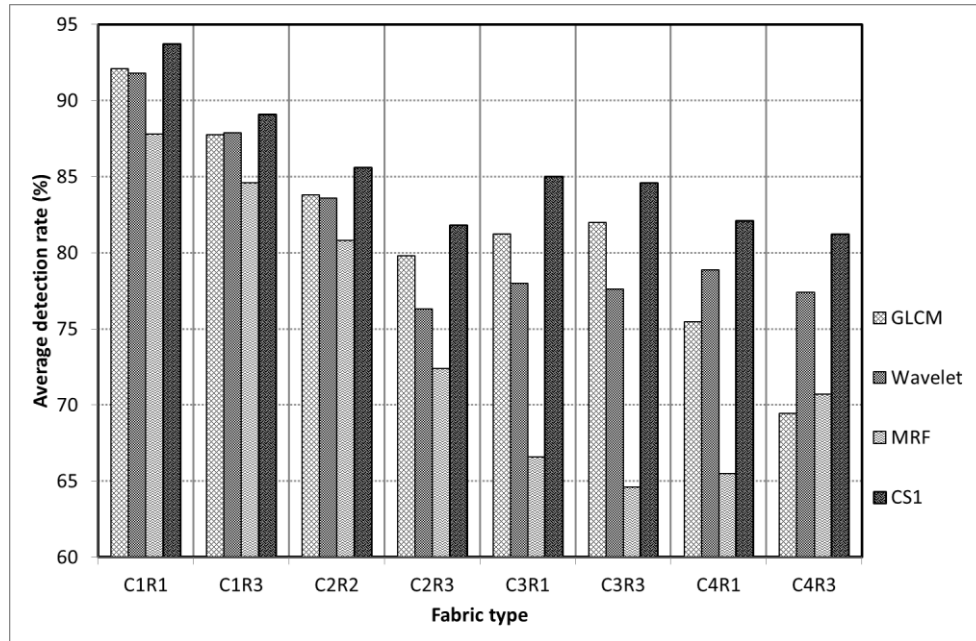


Figure 8-3: Average defect detection performance of combined method CS1 compared to the individual methods

Table 8-3: Average defect detection rate (%) of combined method CS1 compared to the individual GLCM-, wavelet- and MRF-based methods

Fabric type	GLCM	Wavelet	MRF	CS1
C1R1	92.1	91.8	87.8	93.7
C1R3	87.8	87.9	84.6	89.1
C2R2	83.8	83.6	80.8	85.6
C2R3	79.8	76.3	72.4	81.8
C3R1	81.2	78.0	66.6	85.0
C3R3	82.0	77.6	64.6	84.6
C4R1	75.5	78.9	65.5	82.1
C4R3	69.5	77.4	70.7	81.2

8.3 GLCM and MRF features from DTCWT wavelet coefficients (CS2)

8.3.1 Description of the method and the experiment

In the second combination scheme we use the DTCWT wavelet method to decompose the fabric image containing the samples under study into multiscale and multidirectional sub-images. Next the GLCM and MRF features are extracted from the wavelet coefficients corresponding to the samples under study. Then dimensionality reduction using PCA is performed separately on both GLCM and MRF features. The resulting retained principal components are then fed into the MCE-trained Euclidean distance classifier for decision making. The block diagram of Figure 8-4 in illustrates this process.

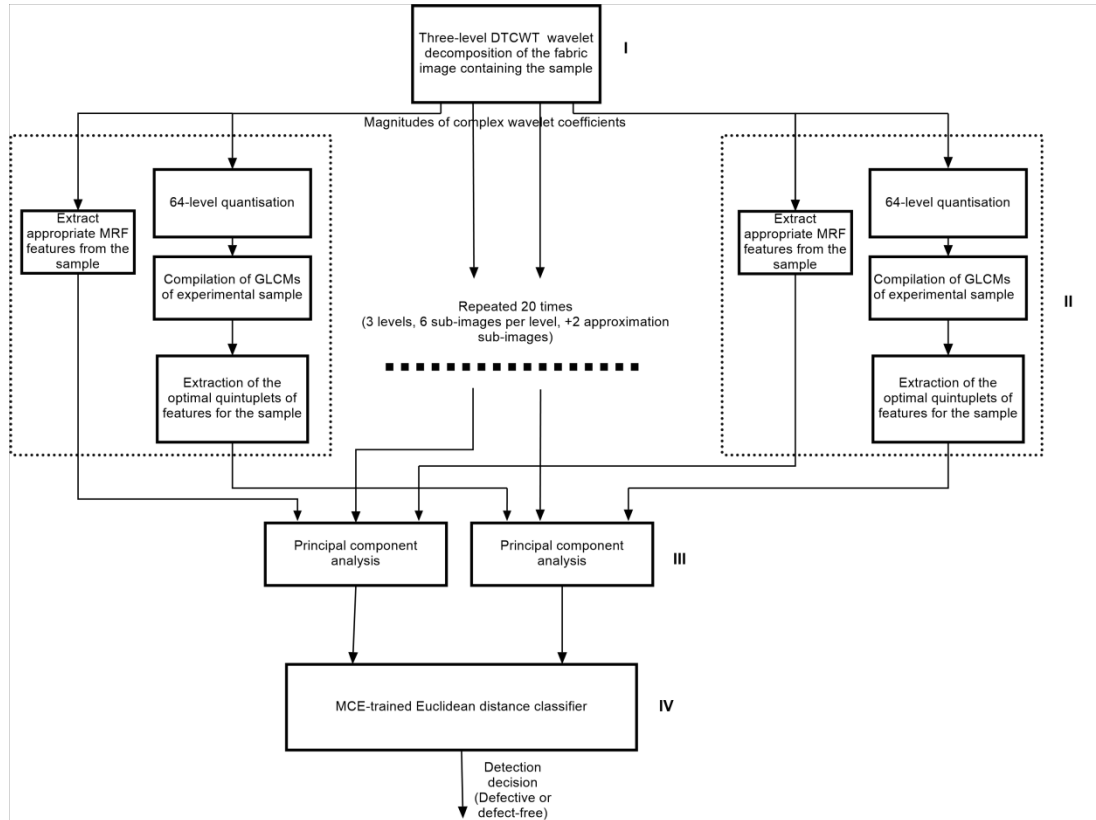


Figure 8-4: Block diagram of the second combination scheme (CS2)

In block I the fabric image containing the sample to be classified as defective or defect-free is submitted to a three-level DTCWT wavelet decomposition using the appropriate wavelet for the fabric type as identified in Subsection 6.8.1. The choice of three levels of decomposition is motivated by the need to have many levels of decompositions for maximising the information from multiple scales, and yet keeping the size of the sample sub-image large enough for GLCM and MRF processing. As the original sample size is 32x32 pixels, and each of the two sides is divided by two by each level of DTCWT decomposition, the size of the sample sub-image at level three is 4x4.

The total number of sub-images generated by the three-level DTCWT wavelet decomposition is twenty, because each level generates six ‘details’ sub-images corresponding to the orientations of -75° , -45° , -15° , $+15^\circ$, $+45^\circ$ and $+75^\circ$. In addition, there are two approximation sub-images that are left after the three-level wavelet decomposition.

The components of block II have been explained previously in chapters 5 and 7. However, it is to be noted that in this case the GLCMs are compiled from the magnitudes of wavelet coefficients which must be properly quantised. In agreement with our observations in Subsection 5.4.3, we quantised them into 64 levels using the uniform quantisation algorithm [47].

The features generated by each of the twenty blocks of type block II are ten from GLCM processing and four, six, eight or twenty (depending on the order of the MRF model, which depends on the fabric type) from the MRF processing. This leads to a total number of features of 280, 320, 360 or 600, and therefore dimensionality reduction is necessary due to the high number of features. To give the same importance to MRF and GLCM features, we used two separate instances of PCA processing (block III), one for each type of feature, and retain the same number of principal components from each of them. Finally, in block IV the MCE-trained Euclidean distance classifier was implemented as previously explained in Subsection 3.5.3.

8.3.2 Results and interpretation

Figure 8-5 and Table 8-4 show the variation of the average defect detection rate for the different types of fabric as the number of the principal components of the GLCM and MRF features increases. We can see from the graph that the average detection rate increases rapidly with the number of principal component and then stabilises. With 76 principal components of GLCM and MRF features, the average detection rate is practically stable. Therefore we chose 76 as the number of principal components of GLCM and MRF features to retain.

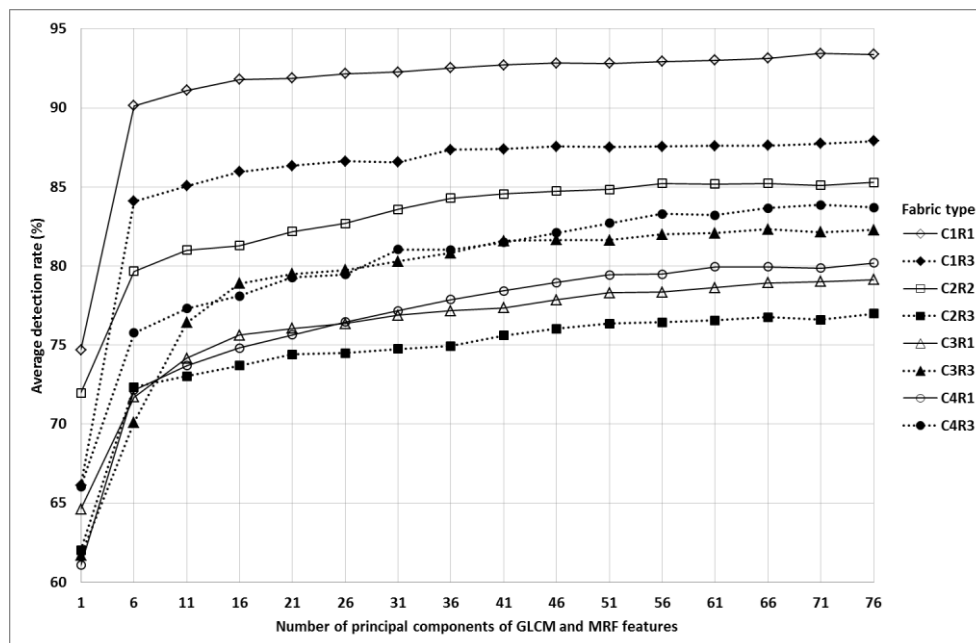


Figure 8-5: Defect detection performance of combined method CS2 for different fabric types as number of principal components of wavelet features changes

Table 8-4: Average defect detection rate (%) of combined method CS2 for different fabric types as number of principal components of GLCM and MRF changes

Number of principal components of GLCM and MRF features	Fabric types							
	C1R1	C1R3	C2R2	C2R3	C3R1	C3R3	C4R1	C4R3
1	74.7	66.1	72.0	62.0	64.6	61.7	61.1	66.0
6	90.1	84.1	79.7	72.3	71.7	70.1	72.2	75.8
11	91.1	85.1	81.0	73.0	74.2	76.4	73.7	77.3
16	91.8	86.0	81.3	73.7	75.6	78.9	74.8	78.1
21	91.9	86.3	82.2	74.4	76.1	79.5	75.7	79.3
26	92.2	86.6	82.7	74.5	76.4	79.7	76.5	79.5
31	92.3	86.6	83.6	74.8	76.9	80.3	77.2	81.0
36	92.5	87.3	84.3	74.9	77.2	80.8	77.9	81.0
41	92.7	87.4	84.6	75.6	77.3	81.6	78.4	81.5
46	92.8	87.6	84.7	76.0	77.8	81.7	78.9	82.1
51	92.8	87.5	84.8	76.3	78.3	81.6	79.4	82.7
56	92.9	87.6	85.2	76.4	78.3	82.0	79.5	83.3
61	93.0	87.6	85.2	76.6	78.6	82.1	80.0	83.2
66	93.1	87.6	85.2	76.8	78.9	82.3	80.0	83.7
71	93.4	87.7	85.1	76.6	79.0	82.1	79.9	83.9
76	93.4	87.9	85.3	77.0	79.1	82.3	80.2	83.7

The comparison of the defect detection performance of CS2 to the individual methods is shown in Figure 8-6 and Table 8-5. The improvement of the defect detection performance varies with the fabric type. There is significant improvement for fabric types C4R3, C1R1, C4R1 and C2R2. There is practically no improvement for fabric type C1R3, for which the detection performance is almost the same as that of GLCM or wavelet alone. Similarly, no improvement is observed for fabric type C3R3, for which the detection performance is almost the same as that of GLCM alone. For fabric types C2R3 and C3R1 there is significant drop in detection rate, as CS2 is outperformed by GLCM alone.

In short the CS2-method should be tested carefully on a particular fabric type before being used, as there is no assurance it would lead to improved performance. However, for most of the fabric types in our dataset the defect detection performance of CS2 was at least the same as the best of the individual methods.

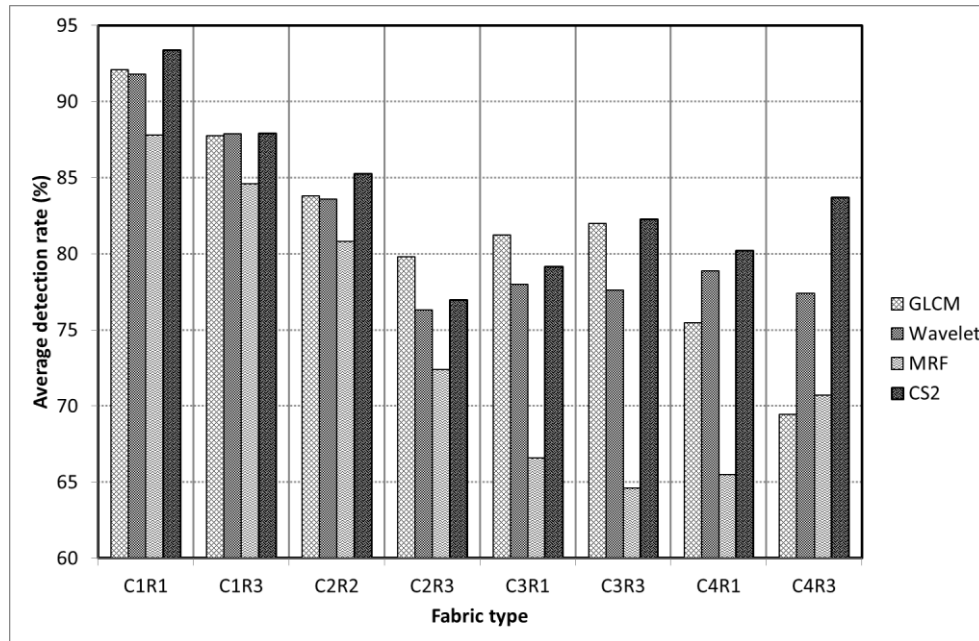


Figure 8-6: Defect detection performance of combined method CS2 compared to the individual methods

Table 8-5: Average defect detection rate (%) of combined method CS2 compared to the individual GLCM-, wavelet- and MRF-based methods

Fabric type	GLCM	Wavelet	MRF	CS2
C1R1	92.1	91.8	87.8	93.4
C1R3	87.8	87.9	84.6	87.9
C2R2	83.8	83.6	80.8	85.3
C2R3	79.8	76.3	72.4	77.0
C3R1	81.2	78.0	66.6	79.1
C3R3	82.0	77.6	64.6	82.3
C4R1	75.5	78.9	65.5	80.2
C4R3	69.5	77.4	70.7	83.7

8.4 GLCM and MRF features from DTCWT reconstructed images (CS3)

8.4.1 Description of the method and the experiment

The third combination scheme, CS3, is similar to the second with the difference that the GLCM and MRF features are not extracted from the wavelet coefficients, but from the fully reconstructed sub-images at different scales and different directions.

The block diagram in Figure 8-7 shows the flow and processing of information for that particular combination scheme. The fabric image containing the sample under study is submitted to a three-level DTCWT wavelet decomposition using the appropriate wavelet filters for the particular fabric type (block I). This gives complex wavelet coefficients in eighteen “details” subbands and two approximation subbands.

Each of the twenty subbands is reconstructed separately (block II). This is accomplished by taking all the wavelet coefficients of the original image, setting to zeros all the wavelet coefficients apart from those that are part of the current subband, and then performing a three-level wavelet reconstruction. The DTCWT wavelet decomposition and reconstruction are done by respectively using the MATLAB functions ‘*cplxdual2*’ and ‘*icplxdual2*’ provided by Polytechnic University, Brooklyn, NY [183].

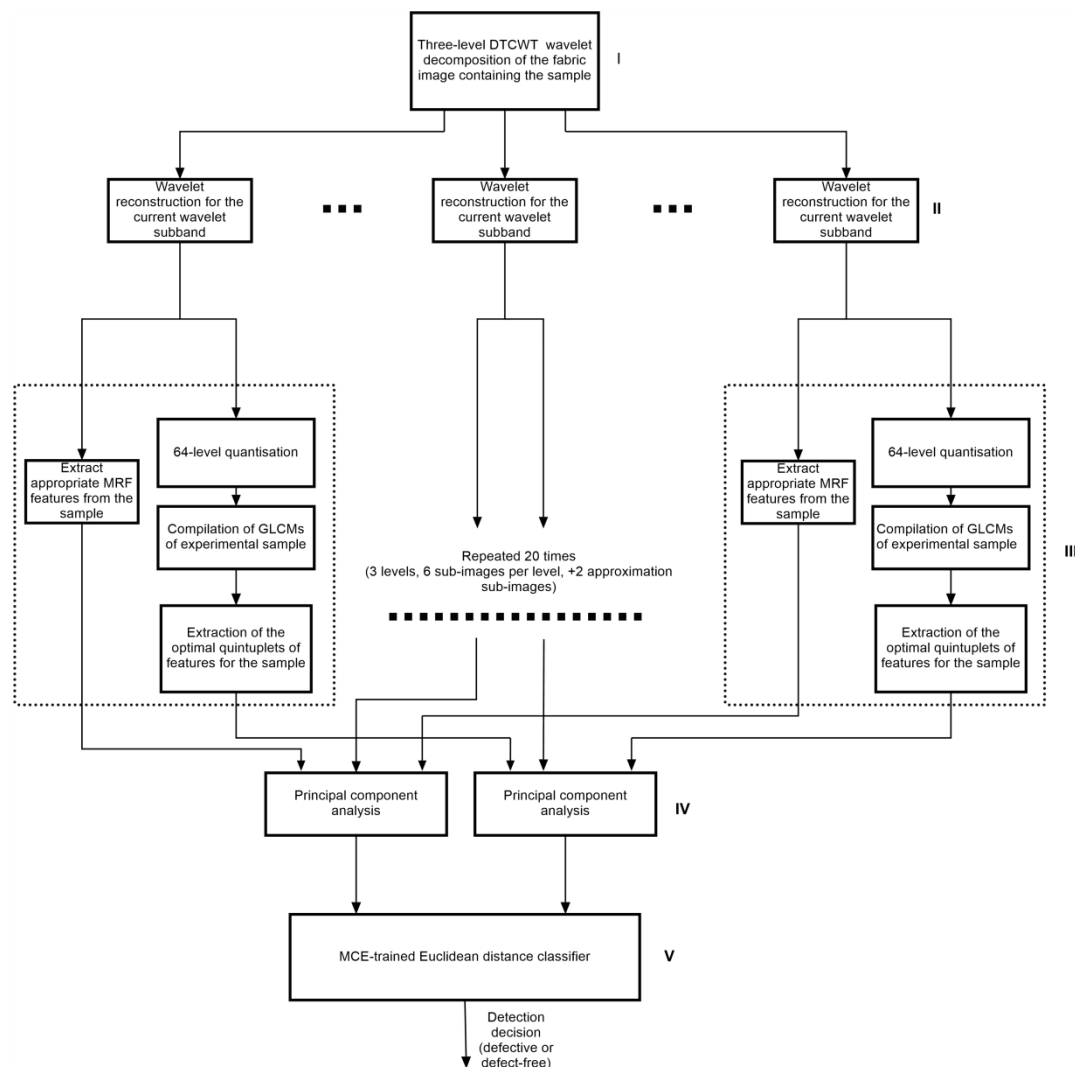


Figure 8-7: Block diagram of the third combination scheme (CS3)

After reconstruction of the sub-images, the appropriate GLCM and MRF features of the sample under study are extracted (block IV). The MRF and GLCM features are then submitted separately to PCA processing (block II), and 76 principal components of each of the two types of features are fed into a MCE-trained classifier for decision making (block V). The steps in blocks III, IV and V are exactly as described for the combination scheme CS2 in Subsection 8.3.1.

8.4.2 Results and interpretation

Figure 8-8 and Table 8-6 show the defect detection performance of the combined method CS3 compared to the individual methods. We observe that for all the fabric types in our dataset the combined method CS3 performs better than any individual method alone. The improvement is significant for all the fabric types in our dataset, apart from C2R3 where the average detection of combined method CS3 is almost the same as GLCM. We can therefore conclude that combined method CS3 effectively improves the defect detection performance.

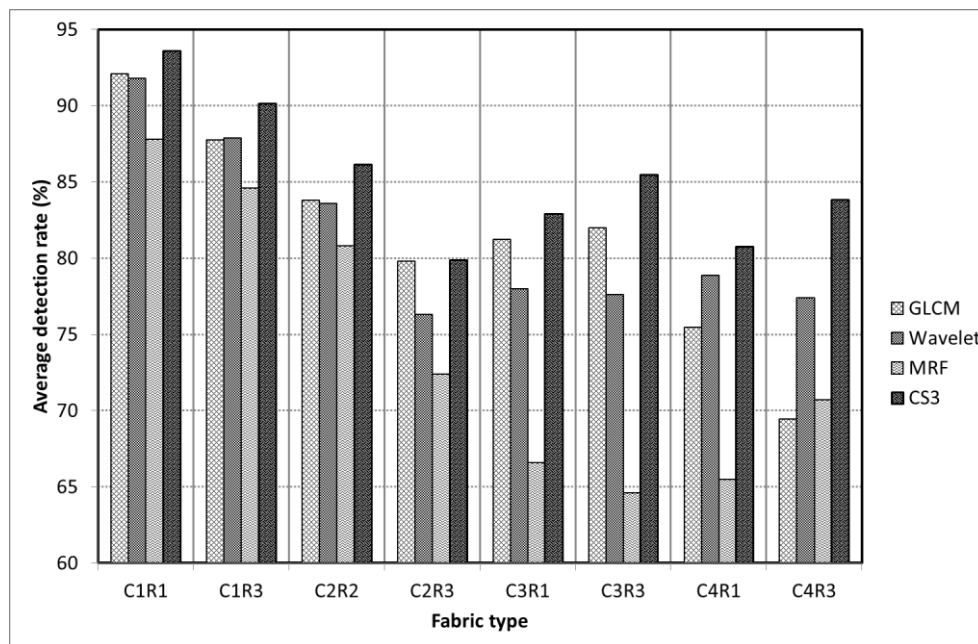


Figure 8-8: Defect detection performance of combined method CS3 compared to the individual methods

Table 8-6: Average defect detection rate (%) of combined method CS3 compared to the individual GLCM-, wavelet- and MRF-based methods

Fabric type	GLCM	Wavelet	MRF	CS3
C1R1	92.1	91.8	87.8	93.6
C1R3	87.8	87.9	84.6	90.1
C2R2	83.8	83.6	80.8	86.1
C2R3	79.8	76.3	72.4	79.9
C3R1	81.2	78.0	66.6	82.9
C3R3	82.0	77.6	64.6	85.5
C4R1	75.5	78.9	65.5	80.7
C4R3	69.5	77.4	70.7	83.8

8.5 Comparison of the three combination schemes

Figure 8-9 and Table 8-7 compare the defect detection performance of combined methods CS1, CS2 and CS3 when the MCE-trained Euclidean distance classifier is used. We recommend the suitable combined method based on the average detection rate and the complexity of the method for each fabric type.

For fabric type C1R1 the combined methods CS1 and CS3 have almost the same average detection rate, while the average detection rate for CS2 is lower. For this fabric type we recommend the combined method CS1 because it is simpler than CS3.

We recommend the combined method CS3 for fabric types C1R3, C2R2 and C3R3 because it performs better than the two others.

For fabric types C2R3, C3R1 and C4R1 the recommended combined method is CS1 because of its higher detection performance and relative simplicity.

Finally, for fabric type C4R3 the average detection rate of CS2 is almost the same as the detection rate of CS3, while that of CS1 is much lower. In this case we recommend CS2 because it is simpler than CS3.

Table 8-8 summarises the recommended combined method for each of the fabric types.

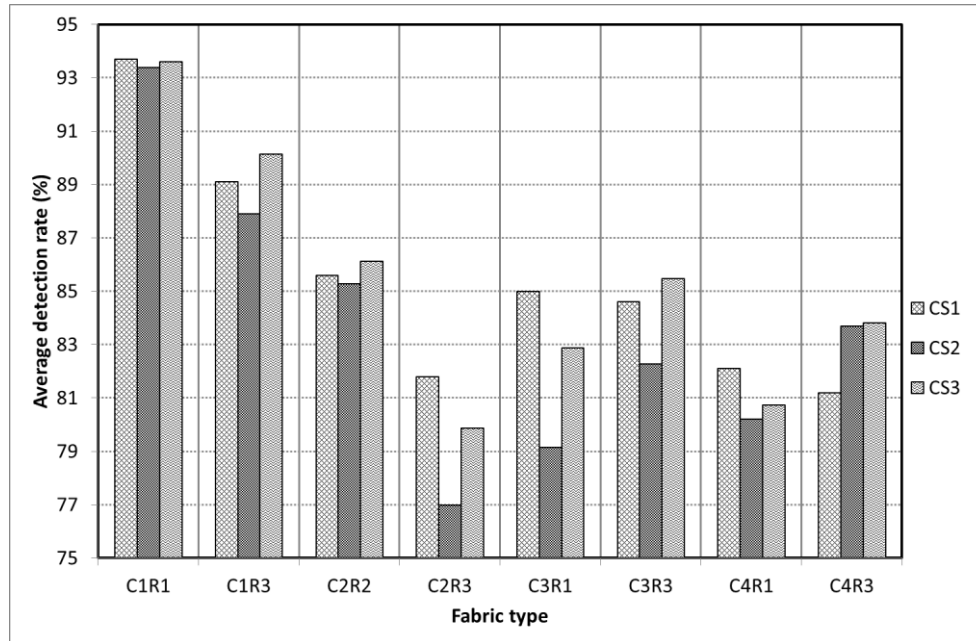


Figure 8-9: Comparison of defect detection performance of the three combined methods

Table 8-7: Compared average defect detection rates (%) of the three combined methods

Fabric type	CS1	CS2	CS3
C1R1	93.7	93.4	93.6
C1R3	89.1	87.9	90.1
C2R2	85.6	85.3	86.1
C2R3	81.8	77.0	79.9
C3R1	85.0	79.1	82.9
C3R3	84.6	82.3	85.5
C4R1	82.1	80.2	80.7
C4R3	81.2	83.7	83.8

Table 8-8: Recommended defect detection combined method for each fabric type

Fabric type	Recommended combined method	Average detection rate (%)
C1R1	CS1	93.7
C1R3	CS3	90.1
C2R2	CS3	86.1
C2R3	CS1	81.8
C3R1	CS1	82.9
C3R3	CS3	85.5
C4R1	CS1	82.1
C4R3	CS2	83.7

8.6 Effect of different classifiers

Thus far all the experiments involving combined methods have been performed using the MCE-trained Euclidean distance classifier. In this section we perform the same experiments, but using the K-nearest-neighbour and the feed-forward neural network classifiers. The objective is to confirm the observations made or to identify any departure from them due to the use of different classifiers. Especially, we want to assure the improvement of the defect detection performance of the combined methods with different classifiers.

8.6.1 Combined schemes using the KNN classifier

To evaluate the performance of the combination schemes using the KNN classifier we performed exactly the same experiments as we did under sections 8.2 through 8.5, but using the K-nearest-neighbour classifier instead of the MCE-trained Euclidean distance classifier. We also changed the set of best features to use those identified for the K-nearest-neighbour classifier as shown in Table 8-9.

Table 8-9: Most effective features for the fabric defect detection methods to be combined when K-nearest-neighbour classifier is used

Fabric type	GLCM	Wavelet				MRF
	Optimal quintuplets of feature set	Wavelet transform type	Wavelet filter length	Number of levels	Feature set	Model order
C1R1	{f2, f6, f10, f11, f14}	DTCWT	4	5	Set c2	1
C1R3	{f5, f7, f8, f10, f11}	DTCWT	4	5	Set c2	1
C2R2	{f2, f6, f8, f11, f14}	DTCWT	4	5	Set c2	1
C2R3	{f1, f2, f11, f12, f13}	DTCWT	4	5	Set c2	1
C3R1	{f2, f4, f9, f11, f13}	DTCWT	10	5	Set c2	1
C3R3	{f1, f7, f8, f11, f14}	DTCWT	10	5	Set c2	2
C4R1	{f4, f8, f9, f11, f14}	UDWT	8	3	Set r2	2
C4R3	{f1, f6, f10, f11, f14}	UDWT	8	3	Set r2	2

The obtained results are shown in Figure 8-10 and Table 8-10. We can see that combined method CS1 improves the defect detection performance for only fabric types C1R1, C2R2 and C4R3. In all the other cases either GLCM or wavelet-based features perform better.

The observations regarding combined methods CS2 and CS3 are that neither of the two improves the detection performance when the K-nearest-neighbour classifier is used.

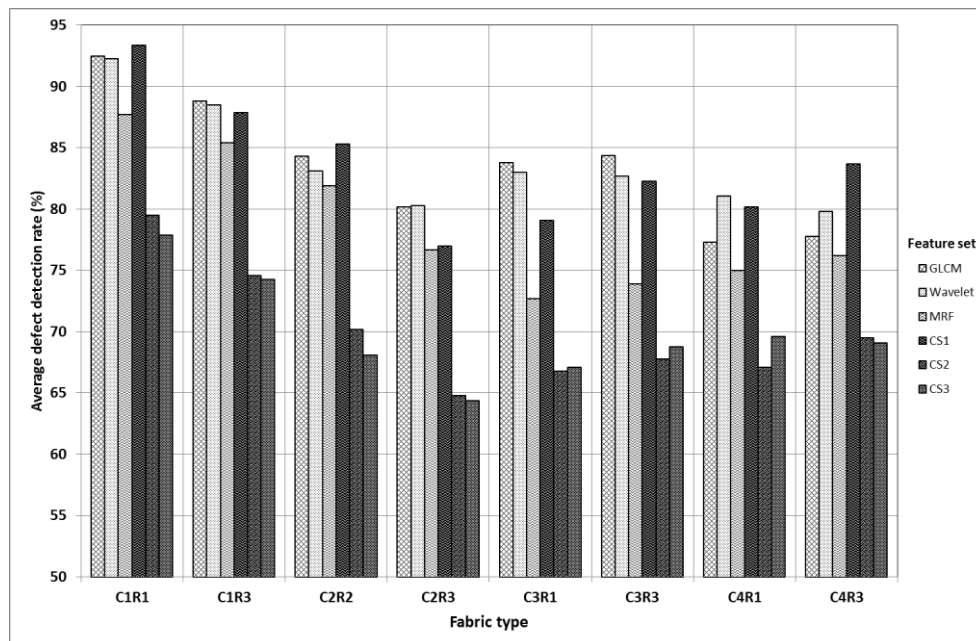


Figure 8-10: Defect detection performance of combined methods CS1, CS2 and CS3 compared to the individual GLCM-, wavelet- and MRF-based methods when the K-nearest-neighbour classifier is used

Table 8-10: Average defect detection rate (%) of combined methods CS1, CS2 and CS3 compared to the individual GLCM-, wavelet- and MRF-based methods when the K-nearest-neighbour classifier

Fabric type	GLCM	Wavelet	MRF	CS1	CS2	CS3
C1R1	92.5	92.3	87.7	93.4	79.5	77.9
C1R3	88.8	88.5	85.4	87.9	74.6	74.3
C2R2	84.3	83.1	81.9	85.3	70.2	68.1
C2R3	80.2	80.3	76.7	77.0	64.8	64.4
C3R1	83.8	83.0	72.7	79.1	66.8	67.1
C3R3	84.4	82.7	73.9	82.3	67.8	68.8
C4R1	77.3	81.1	75.0	80.2	67.1	69.6
C4R3	77.8	79.8	76.2	83.7	69.5	69.1

8.6.2 Combined schemes using the feed-forward neural network classifier

We evaluate the performance of the combination schemes using the feed-forward neural network by performing the same experiments as we did under Sections 8.2 through 8.5 but using the feed-forward neural network classifier instead of the MCE-trained Euclidean distance classifier. We also change the GLCM-, wavelet- and MRF-based best features to use those identified specifically for the feed-forward neural network classifier as shown in Table 8-11. For CS1 we used twenty principal components of the wavelet-based features, while for CS2 and CS3 we used 60 principal components of GLCM and MRF features.

Table 8-11: Most effective features for fabric defect detection methods to be combined when the feed-forward neural classifier is used

Fabric type	GLCM	Wavelet				MRF
	Optimal nonuplets of feature set	Wavelet transform type	Wavelet filter length	Number of levels	Feature set	Model order
C1R1	{f2, f5, f6, f7, f9, f10, f11, f13, f14}	DTCWT	4	5	Set c2	1
C1R3	{f1, f2, f5, f7, f8, f9, f10, f11, f14}	DTCWT	4	5	Set c2	1
C2R2	{f1, f2, f3, f4, f5, f10, f11, f12, f14}	DTCWT	4	5	Set c2	1
C2R3	{f2, f4, f5, f7, f9, f10, f11, f12, f14}	DTCWT	6	5	Set c2	4
C3R1	{f1, f2, f6, f8, f9, f10, f11, f12, f14}	DTCWT	10	5	Set c2	1
C3R3	{f2, f3, f4, f5, f6, f8, f10, f12, f13}	DTCWT	10	5	Set c2	7
C4R1	{f3, f5, f6, f7, f8, f9, f10, f11, f14}	UDWT	6	3	Set r2	7
C4R3	{f2, f4, f6, f7, f9, f10, f11, f12, f14}	UDWT	6	3	Set r2	8

The obtained results are shown in Figure 8-11 and Table 8-12. These results show that combined method CS1 significantly improves the defect detection performance for all the fabric types in our dataset apart from the fabric type C3R3 for which CS1 is slightly outperformed by GLCM- and wavelet-based features.

Neither CS2 nor CS3 improves the defect detection performance when the feed-forward neural network classifier is used. That observation is valid for all the fabric types in our dataset although CS3 performs equally well as GLCM features for the fabric type C1R1.

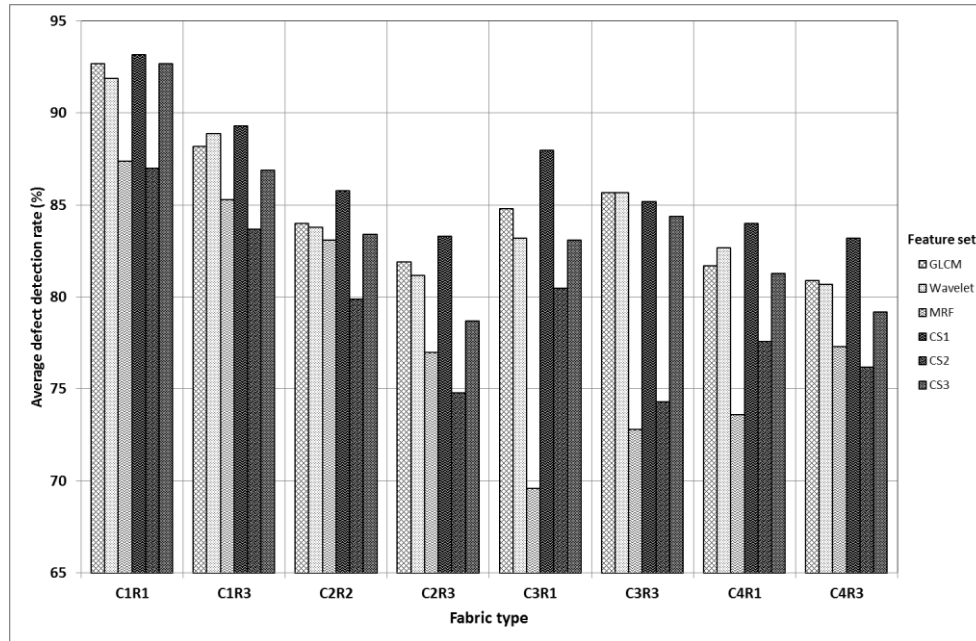


Figure 8-11: Defect detection performance of combined methods CS1, CS2 and CS3 compared to the individual GLCM-, wavelet- and MRF-based methods when the feed-forward neural network classifier is used

Table 8-12: Average defect detection rate (%) of combined methods CS1, CS2 and CS3 compared to the individual GLCM-, wavelet- and MRF-based methods when the feed-forward neural classifier is used

Fabric type	GLCM	Wavelet	MRF	CS1	CS2	CS3
C1R1	92.7	91.9	87.4	93.2	87.0	92.7
C1R3	88.2	88.9	85.3	89.3	83.7	86.9
C2R2	84.0	83.8	83.1	85.8	79.9	83.4
C2R3	81.9	81.2	77.0	83.3	74.8	78.7
C3R1	84.8	83.2	69.6	88.0	80.5	83.1
C3R3	85.7	85.7	72.8	85.2	74.3	84.4
C4R1	81.7	82.7	73.6	84.0	77.6	81.3
C4R3	80.9	80.7	77.3	83.2	76.2	79.2

8.7 Defect classification using the combined methods

In this section we compare the defect classification performance achieved with the features from the combination schemes CS1, CS2 and CS3 to that achieved by GLCM-, wavelet- or MRF-based features individually. This will allow assessing the effectiveness of the combined schemes for fabric defect classification.

Figure 8-12 and Table 8-13 show the obtained results when the MCE-trained Euclidean distance classifier is used. They show that combined method CS1 significantly improved the

defect classification rates for all the fabric types in our dataset apart from the fabric types C2R3 and C4R1. Table 8-14 shows the confusion matrices that provide details of the improvement for fabric type C1R1. Combined method CS2 improves the defect detection rates of individual methods for fabric types C3R1, C3R3 and C4R3. Combined method CS3 only improves the defect classification rates for fabric types C4R3 and C3R3. It is also interesting to notice that all three combined methods outperform any of the individual methods for fabric types C3R3 and C4R3, while none do for the fabric types C2R3 and C4R1.

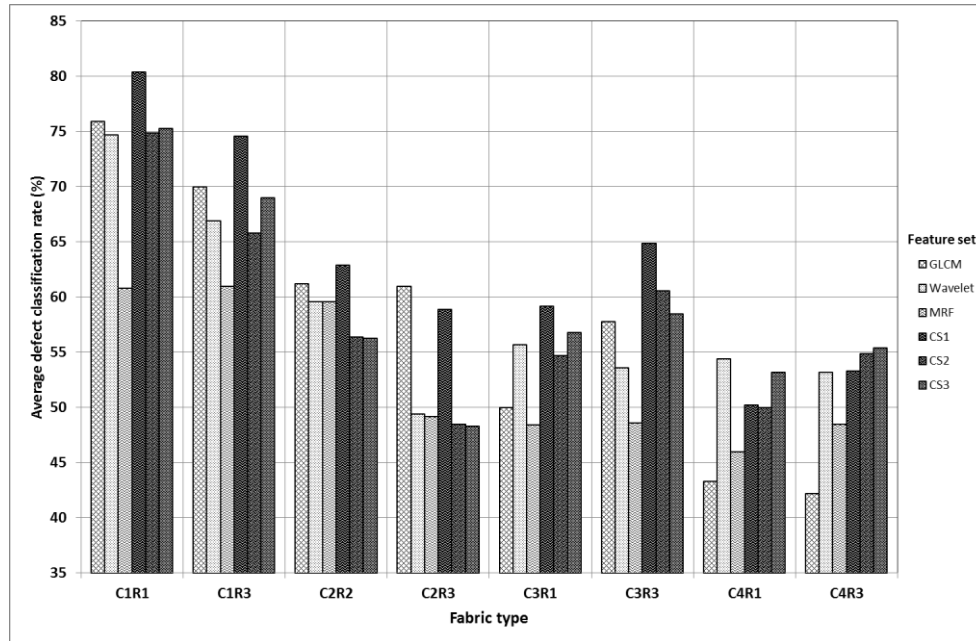


Figure 8-12: Defect classification performance of combined methods CS1, CS2 and CS3 compared to the individual GLCM-, wavelet- and MRF-based methods when the MCE-trained Euclidean distance classifier is used

Table 8-13: Average defect classification rate (%) of combined methods CS1, CS2 and CS3 compared to the individual GLCM-, wavelet- and MRF-based methods when the MCE-trained Euclidean distance classifier is used.

Fabric type	GLCM	Wavelet	MRF	CS1	CS2	CS3
C1R1	75.9	74.7	60.8	80.4	74.9	75.3
C1R3	70.0	66.9	61.0	74.6	65.8	69.0
C2R2	61.2	59.6	59.6	62.9	56.4	56.3
C2R3	61.0	49.4	49.2	58.9	48.5	48.3
C3R1	50.0	55.7	48.4	59.2	54.7	56.8
C3R3	57.8	53.6	48.6	64.9	60.6	58.5
C4R1	43.3	54.4	46.0	50.2	50.0	53.2
C4R3	42.2	53.2	48.5	53.3	54.9	55.4

Table 8-14: Confusion matrices comparing defect classification for fabric type C1R1 using GLCM features and CS1 combined features using the MCE-trained Euclidean distance classifier
(There are less misclassified samples i.e. off-diagonal elements in the confusion matrix (b))

GLCM features (a)							CS1 Combined features (b)						
		when their true class is							when their true class is				
		E0	E1	E2	E3	E4			E0	E1	E2	E3	E4
Classified as	E0	4532	4	153	62	240	Classified as	E0	4669	10	158	18	136
	E1	42	276	6	45	49		E1	13	305	20	37	43
	E2	144	28	525	35	352		E2	245	5	672	30	132
	E3	102	62	48	151	88		E3	129	30	41	202	49
	E4	218	62	144	46	582		E4	260	37	140	37	578
Classification rate: 75.9%							Classification rate: 80.4%						

The obtained results when the K-nearest-neighbour classifier is used can be read from Figure 8-13 and Table 8-15. They show that combined method CS1 significantly improves the defect classification rates for all the fabrics in our dataset. They also show that CS2 and CS3 do so for only fabric type C4R3. For all the other fabric types CS2 and CS3 are outperformed by either GLCM, wavelet-based method, or both.

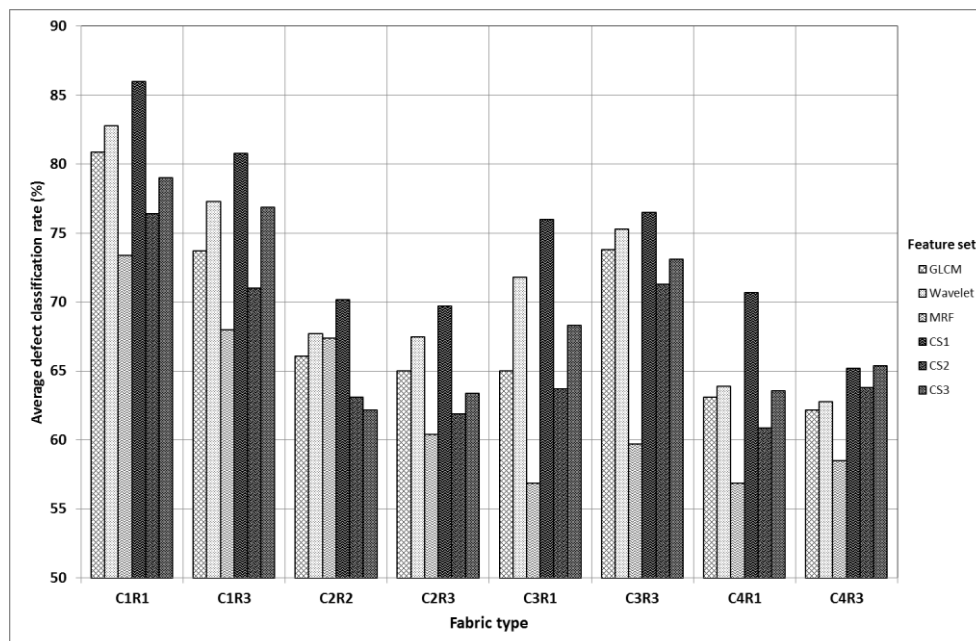


Figure 8-13: Defect classification performance of combined methods CS1, CS2 and CS3 compared to the individual GLCM-, wavelet- and MRF-based methods when the K-nearest-neighbour classifier is used

Table 8-15: Average defect classification rate (%) of combined methods CS1, CS2 and CS3 compared to the individual GLCM-, wavelet- and MRF-based methods when K-nearest-neighbour classifier is used.

Fabric type	GLCM	Wavelet	MRF	CS1	CS2	CS3
C1R1	80.9	82.8	73.4	86.0	76.4	79.0
C1R3	73.7	77.3	68.0	80.8	71.0	76.9
C2R2	66.1	67.7	67.4	70.2	63.1	62.2
C2R3	65.0	67.5	60.4	69.7	61.9	63.4
C3R1	65.0	71.8	56.9	76.0	63.7	68.3
C3R3	73.8	75.3	59.7	76.5	71.3	73.1
C4R1	63.1	63.9	56.9	70.7	60.9	63.6
C4R3	62.2	62.8	58.5	65.2	63.8	65.4

Finally, Figure 8-14 and Table 8-16 show the obtained results when the feed-forward neural network classifier is used. We can see that CS1 improves the defect classification rates for fabric types C2R2, C3R1, C4R1 and C4R3, while CS2 and CS3 do not improve classification rates for any of the fabric types. However, CS3 achieves almost the same performance as wavelet-based features (the best for that fabric type) for C2R2.

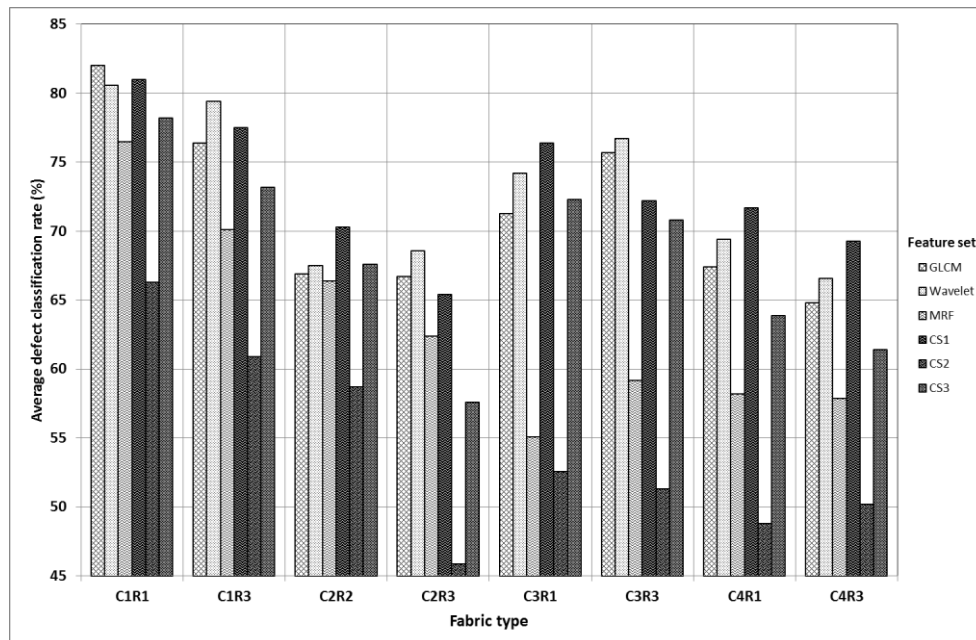


Figure 8-14: Defect classification performance of combined methods CS1, CS2 and CS3 compared to the individual GLCM-, wavelet- and MRF-based methods when the feed-forward neural network classifier is used

Table 8-16: Average defect classification rate (%) of combined methods CS1, CS2 and CS3 compared to the individual GLCM-, wavelet- and MRF-based methods when feed-forward neural network classifier is used

Fabric type	GLCM	Wavelet	MRF	CS1	CS2	CS3
C1R1	82.0	80.6	76.5	81.0	66.3	78.2
C1R3	76.4	79.4	70.1	77.5	60.9	73.2
C2R2	66.9	67.5	66.4	70.3	58.7	67.6
C2R3	66.7	68.6	62.4	65.4	45.9	57.6
C3R1	71.3	74.2	55.1	76.4	52.6	72.3
C3R3	75.7	76.7	59.2	72.2	51.3	70.8
C4R1	67.4	69.4	58.2	71.7	48.8	63.9
C4R3	64.8	66.6	57.9	69.3	50.2	61.4

In short, for fabric defect classification only combined method CS1 shows good results. Methods CS2 and CS3 improve the defect classification results only in a few cases of fabric types and classifiers. However, these two combined methods are based on sound grounds of features extraction from multiple scales. This implies that more research could lead to their improved performance for both fabric defect detection and classification.

8.8 Summary

In this chapter we devised combination schemes of the best GLCM-, wavelet- and MRF-based features for different types of fabrics identified in chapters 5, 6 and 7 respectively and then tested their defect detection and classification performance using three different types of classifiers, namely (i) the MCE-trained Euclidean distance classifier, (ii) the K-nearest-neighbour classifier and (iii) the feed-forward neural network classifier.

In the first combination scheme (CS1), we pooled together the best features identified in previous chapters and fed them into the classifier. Due to the higher number of wavelet-based features, we submitted them to dimensionality reduction using PCA before feeding them into a classifier along with the GLCM- and MRF-based features.

In the second combination scheme (CS2), we submitted the fabric images to a three-level DTCWT decomposition and then the best GLCM- and MRF-based features were extracted from the magnitude of wavelet coefficients. We submitted the resulting GLCM- and MRF-based features separately to dimensionality reduction using PCA and then fed the resulting features into a classifier.

In the third combination scheme (CS3), we submitted the fabric images to a three-level DTCWT decomposition, then the sub-images for each of the three levels and each of the six directions were reconstructed from the sub-image wavelet coefficients by inverse DTCWT transform. Then the best GLCM- and MRF-based features were extracted from the resulting twenty images for each fabric image. The obtained GLCM- and MRF-based features were then submitted separately to dimensionality reduction using PCA before feeding them into a classifier.

The fabric defect detection results obtained using the MCE-trained classifier show that methods CS1 and CS3 perform better than any individual method (i.e. GLCM-, wavelet- or MRF-based) for all the fabric types in our dataset. Method CS2 performed at least equally well as the best individual feature set, apart for fabric types C2R3 and C3R1 for which GLCM features performed better.

However, the improvement for the fabric defect detection may depend on the classifier used. For example, when the K-nearest-neighbour classifier is used only CS1 improves the fabric defect detection performance. This improvement is observed for only a few types of fabrics. When the feed-forward neural network classifier is used CS1 improves the defect detection performance for most of the types of fabrics, but CS2 and CS3 do not.

Defect classification results using the combined methods showed that only combined method CS1 shows good results. Methods CS2 and CS3 improve the defect classification results only in a few cases of fabric types and classifiers.

Chapter 9: Conclusions and future work

9.1 Conclusions

In this thesis we studied the detection and classification performance of three different methods and then combined them for improved performance. The concerned methods are (i) the grey level co-occurrence matrix (chapter 5), (ii) the wavelet transform (chapter 6) and (iii) the Markov random field models (chapter 7).

To this effect we studied the methods individually in order to identify the most effective features of each of them and then the identified features were combined (chapter 8). To evaluate the validity of the defect detection and classification observations across classifiers, the methods were applied using three different classifiers, namely (i) the MCE-trained Euclidean distance classifier, (ii) the K-nearest-neighbour classifier and (iii) the feed-forward neural network classifier.

9.1.1 Conclusions regarding GLCM features

The objective was to find the most effective features for fabric defect detection and classification. The other objective was to identify the values of parameters of GLCM feature extraction that allow alleviating the related computational load.

In this respect we set out to find the optimal number of GLCM features among the fourteen proposed by Haralick et al. [31]. Once the number of optimal features was determined, we identified the actual features for each of the fabric types in our dataset.

We made the following observations:

- (i) The optimal number of features is classifier-dependent and for the classifiers we used we found that it is three for the ML-trained Euclidean distance classifier, five for both the MCE-trained Euclidean distance and the KNN (K=3) classifiers, and nine for the feed-forward neural network classifier.
- (ii) The actual features for different fabric types in our dataset were identified using the procedure that we proposed.
- (iii) Since reducing the number of quantisation levels of the original image reduces the time required for GLCM feature extraction, we studied the variation of the defect detection accuracy with regard to the number of quantisation levels. We

found that 64 grey levels were optimal for accelerating the feature extraction without affecting the defect detection accuracy too much.

- (iv) The best intersample distance for compiling the co-occurrence matrices was found to be 1 pixel for the fabric types we used.
- (v) The feed-forward neural network classifier provided the best defect detection and classification rates among the classifiers we used.
- (vi) We proposed the modified GLCHS algorithm for faster GLCM feature extraction for implementation in MATLAB.

9.1.2 Conclusions regarding wavelet features

To determine the most effective wavelet-based features we compared the defect detection and classification performance of features derived from two shift-invariant wavelet transforms: (i) the undecimated discrete wavelet transform, and (ii) the dual-tree complex wavelet transform. For each of the two types of wavelet transform we investigated the defect detection performance of derived features with regard to the wavelet size of support and the number of wavelet decomposition levels.

We made the following observations:

- (i) The defect detection rate of the UDWT-based features tends to decrease as the wavelet size of support increases.
- (ii) For the DTCWT-based features the effect of the wavelet size of support is less pronounced than that of the UDWT-based features. However, a slight increasing trend is noticed.
- (iii) General observations (i) and (ii) are not uniform for different fabric types.
- (iv) These observations seem to be classifier-invariant based on the results from the three different classifiers that we used; the MCE-trained Euclidean distance classifier, the KNN classifier (K=3) and the feed-forward neural network classifier.
- (v) The feed-forward neural network classifier provided the best detection and classification rates among the classifiers we used.

We compared the defect detection and classification performance of UDWT-based features to that of the DTCWT-based features for different fabrics and made the following observations:

- (i) DTCWT-based features perform better than UDWT-based ones for most of the fabric types, apart from the printed fabrics with no apparent periodicity in their texture for which the UDWT-based features performed better.
- (ii) For each fabric type in our dataset we identified the best wavelet features in terms of the type of wavelet transform (UDWT or DTCWT) and the wavelet size of support. Short wavelets seem to perform better than long ones and DTCWT generally performs better than UDWT.

9.1.3 Conclusions regarding MRF features

We studied the defect detection and classification performance of features derived from Gaussian Markov field models of fabric images. We specifically investigated the variation of the defect detection rate of the derived features as the order of the model varied from one through nine for different fabric types. We made the following observations:

- (i) The texture features extracted using the first order MRF model are the most effective for defect detection in fine fabrics (C1R1, C1R3, C2R2 and C2R3).
- (ii) For these fine fabrics the defect detection performance tends to decrease as the order of the model increases.
- (iii) For coarser fabrics (C3R1, C3R3, C4R1 and C4R3) the defect detection performance seems either to have low dependence with respect to the order of the model or to slightly increase as the order of the model increases.
- (iv) When used for defect classification, the features extracted from low-order MRF models perform the best, especially for the fine fabrics.
- (v) For defect detection from MRF-based features the three classifiers we used seemed to perform equally well, while for defect classification the feed-forward neural network classifier performed better than the K-nearest-neighbour classifier, and the K-nearest-neighbour classifier better than the MCE-trained Euclidean distance classifier.

9.1.4 Conclusions regarding combination schemes

We combine the best identified GLCM-, wavelet- and MRF-based features using three combination schemes – these constitute new defect detection and classification methods.

In CS1 we pool together the most effective GLCM-, wavelet- and MRF-based features identified for each fabric type in our dataset. We submit the wavelet features to PCA dimensionality reduction, then feed the resulting features, along with the GLCM- and MRF-based features, into a single classifier (case of defect detection) or into a cascade of two classifiers (case of defect classification). The resulting defect detection and classification methods lead to improved defect detection and classification performance in most of the cases (combinations fabric type-classifier).

In CS2 we submit the fabric images to a three-level DTCWT wavelet decomposition. We then calculate the magnitude of the resulting complex wavelet coefficients. Then the GLCM and MRF features are extracted from the resulting sub-images. After the PCA dimensionality reduction of the two sets of features they are fed together into a single classifier or a cascade of two classifiers. The resulting defect detection method generally improves the defect detection performance when the MCE-trained Euclidean distance classifier is used, but fails to do so when the K-nearest-neighbour classifier or the feed-forward neural network are used. The resulting defect classification method leads to improved defect classification only for a few cases of printed fabrics with no apparent periodicity.

In CS3 we also submit the fabric images to a three-level DTCWT wavelet decomposition. We then reconstruct the individual sub-images from the complex wavelet coefficients using the DTCWT inverse wavelet transform. The GLCM and MRF features are then extracted from each of the resulting sub-images. After the PCA dimensionality reduction of the two sets of features they are fed into a single classifier or a cascade of two classifiers for defect detection or defect classification respectively. The resulting defect detection method improves the defect detection performance for all the fabric types in our dataset when the MCE-trained Euclidean distance classifier is used, but fails to do so when either the K-nearest-neighbour classifier or the feed-forward neural network is used. The resulting defect classification method leads to improved defect classification only for a few cases of printed fabrics with no apparent periodicity (C4R3), and for cases of fabrics with clearly visible periodicity when a MCE-trained Euclidean distance classifier is used.

9.2 Recommendations for future work

Combined methods CS2 and CS3 lead to improvement in fabric defect detection when a Euclidean distance classifier is used, but fail to do so when the K-nearest-neighbour classifier ($K=3$) and the feed-forward neural network classifiers are used. In most cases the two combined methods also fail to improve the defect classification performance. Since they are based on sound grounds of features extraction from multiple scales and multiple directions, research focusing on finding why they perform poorly could be conducted to realise improvement.

Combining different methods may lead to improved defect detection and classification rates, but may also lead to greater complexity in the system. The unwanted consequence is the increased computational load. Research focusing on the effect of feature combination on the computation time and the related solutions involving massively parallel algorithms can be undertaken. This is relevant as defect detection and classification algorithms are well suited to parallel computing, and the future of computing performance seems to be parallel processing rather than increasing clock frequencies [188]. This promises cheap heavily parallel computer systems.

The double density dual-tree wavelet transform [189] is also almost shift-invariant and has more free parameters than the dual-tree complex wavelet transform. Future research could seek a way of designing an adaptive double density dual-tree wavelet transform that would optimise the detection rate for a particular type of fabric.

The fabric types from categories C3 and C4, with coarser texture than fabric types from categories C1 and C2, showed consistently low defect detection and classification rates. This was due to the fact that the assumption of the texture uniformity within the study window is less valid for these types of fabrics. Research focusing on the modelling of such fabrics using hierarchical Markov random fields [109], modelling micro textures within macro textures, may lead to better defect detection and classification rates.

Bibliography

- [1] A.C. Cohen, I. Johnson, "JJ Pizzuto's fabric science," Fairchild Books, 2010.
- [2] S. J. Kadolph, "Textiles," Pearson Prentice Hall, 2010.
- [3] K. Srinivasan, P.H. Dastor, P. Radhakrishnaihan, S. Jayaraman, "FDAS: A Knowledge-based Frame-detection work for Analysis of Defects in Woven textile Structures," *J. Text. Inst.*, vol. 83, no. 3, pp. 431-447, 1992.
- [4] J. L. Dorrity, G. Vachtsevanos, W. Jasper, "Real-time Fabric Defect Detection and Control in Weaving Processes," *National Textile Center Annual Report*, pp. 113-122, 1996.
- [5] R. Dewan, M. Aggarwal, "Machine Vision based Detection of Defects in Textile (Fabric) Material," *IOSR Journal of Engineering*, vol.2, Issue 6, pp.16-19, 2012.
- [6] C.S. Cho, B.M. Chung, M.J. Park, "Development of real-time vision based fabric inspection system," *IEEE Transactions on Industrial Electronics*, vol. 52, no. 4, pp. 1073-1079, 2005.
- [7] K.V. Kumar, U. S. Ragupathy, "An Intelligent Scheme for Fault Detection in Textile Web Materials," *International Journal of Computer Applications*, vol. 46, no.10, pp. 24-29, 2012.
- [8] S. Ozdemir, A. Ercil, "Markov random fields and Karhunen-Loève transforms for defect inspection of textile products," In *Emerging Technologies and Factory Automation, 1996. EFTA'96. Proceedings., 1996 IEEE Conference on*, vol. 2, pp. 697-703. IEEE, 1996.
- [9] A.S. Malek, "Online fabric inspection by image processing technology," PhD dissertation, Université de Haute Alsace-Mulhouse, 2012.
- [10] A. Kumar, "Computer-Vision-Based Fabric Defect Detection: A Survey," *IEEE Transactions on Industrial Electronics*, vol. 55, no. 1, pp. 348-363, 2008.
- [11] R.S. Sabeenian, M. E. Paramasivam, "Defect detection and identification in textile fabrics using Multi Resolution Combined Statistical and Spatial Frequency Method," In *Advance Computing Conference (IACC), 2010 IEEE 2nd International*, pp. 162-166. IEEE, 2010.
- [12] X. Z. Yang, G. K. H Pang, N.H.C. Yung, "Fabric Defect Classification Using Wavelet Frames and Minimum Classification Error Training," *IEEE 2002 37th IAS Industry Applications Conference Meetings*, vol.1, pp. 290-296, 2002.
- [13] S. Cohen, "On-loom fabric inspection system and method." WIPO Patent Application PCT/IB2012/051613, filed April 2, 2012.
- [14] A. Dockery, "Automated fabric inspection: assessing the current state of the art," *Techexchange.com*, 2001.
- [15] BMSVision, "Cyclops brochure", 2012.
- [16] X.Z. Yang, "Discriminative fabric defect detection and classification using adaptive wavelet," PhD thesis, The University of Hong Kong, 2003.

- [17] P.M. Mahajan, S.R. Kolhe, P.M. Patil, "A review of automatic fabric defect detection techniques," *Advances in Computational Research*, vol. 1, no. 2, pp. 18-29, 2009.
- [18] S. Özdemir, A. Baykut, R. Meylani, A. Ercil, A. Ertuzun, "Comparative evaluation of texture analysis algorithms for defect inspection of textile products," *IEEE Proceedings Fourteenth International Conference on Pattern Recognition*, vol. 2, pp. 1738–1740, 1998.
- [19] G. Fabrizio, G.B. Francesco, V. Gianni, "A contrast-based approach to the identification of texture faults," *International Journal of Pattern Recognition and Artificial Intelligence (IJPRAI)*, vol. 16, no. 2, pp.193-214, 2002.
- [20] F. S. Cohen, Z. Fan, S. Attali, "Automated inspection of textile fabrics using textural models," *IEEE Trans. Patt. Anal. Machine Intell.*, vol. 13, no. 8, pp. 803-808, 1991.
- [21] A. Latif-Amet, A. Ertüzün, A. Erçil, "An efficient method for texture defect detection: subband domain co-occurrence matrices," *Image and Vision Computing*, vol. 18, no. 6, pp. 543-553, 2000.
- [22] S. Arivazhagan, L. Ganesan, "Texture segmentation using wavelet transform," *Pattern Recognition Letters*, vol. 24, no. 16, pp. 3197-3203, 2003.
- [23] A. Latif-Amet, A. Ertuzun, A. Ercil, "Texture defect detection using subband domain co-occurrence matrices," in: *Proceedings on IEEE Southwest Symposium on Image Analysis and Interpretation*, pp. 205–210, 1998.
- [24] H. Y. Ngan, G. K. Pang, N. H. Yung, "Automated fabric defect detection—A review," *Image and Vision Computing*, vol. 29, no. 7, pp. 442-458, 2011.
- [25] A. Abouelela, I. Abbas, S. Nassar, "A statistical approach for textile fault detection," In *Systems, Man, and Cybernetics, 2000 IEEE International Conference on*, vol. 4, pp. 2857-2862, 2000.
- [26] T.S. Newman, A.K. Jain, "A survey of automated visual inspection," *Computer vision and image understanding*, vol. 61, no. 2, pp. 231-262, 1995.
- [27] H. M. Elragal, "Neuro-Fuzzy Fabric Defect Detection and Classification for Knitting Machine," In *Radio Science Conference, 2006. NRSC 2006. Proceedings of the Twenty Third National*, pp. 1-8. IEEE, 2006.
- [28] H. Balakrishnan, S. Venkataraman, S. Jayaraman, "FDICS: A Vision-based System for the Identification and Classification of Fabric Defects," *Journal of the Textile Institute*, vol. 89, no. 2, pp. 365-380, 1998.
- [29] C. Neubauer, "Segmentation of defects in textile fabric," In *Pattern Recognition, Conference A: Computer Vision and Applications, Proceedings., 11th IAPR International Conference on*, vol. 1, pp. 688-691. IEEE, 1992.
- [30] A. Monadjemi, "Towards efficient texture classification and abnormality detection," PhD dissertation, University of Bristol, 2004.

- [31] R.M. Haralick, K. Shanmugam, L. Dinstein, "Textural features for image classification," *IEEE Trans. Syst., Man., Cybern.*, vol. SMC-3, pp.610-621, 1973.
- [32] A. Bodnarova, J.A. Williams, M. Bennamoun, K.K. Kubik, "Optimal textural features for flaw detection in textile materials," in: *Proceedings on IEEE TENCON'97 Conference*, Brisbane (Aus.), pp.307–310, 1997.
- [33] I.S. Tsai, C.H. Lin, J.J. Lin. "Applying an artificial neural network to pattern recognition in fabric defects," *Textile Research Journal*, vol. 65, no. 3, pp. 123-130, 1995.
- [34] A. Bodmarova, M. Bennamoun, K.K. Kubik, "Suitability analysis of techniques for flaw detection in textiles using texture analysis," *Pattern Analysis & Applications*, vol. 3, no. 3, pp. 254-266, 2000.
- [35] V. Murino, M. Bicego, I.A. Rossi, "Statistical classification of raw textile defects," In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, vol. 4, pp. 311-314. IEEE, 2004.
- [36] C. Kwak, J.A. Ventura, K. Tofang-Sazi, "A neural network approach for defect identification and classification on leather fabric," *Journal of Intelligent Manufacturing*, vol. 11, no. 5, pp. 485-499, 2000.
- [37] C.F.J. Kuo, C.Y. Shih, C.E. Ho, K.C. Peng, "Application of computer vision in the automatic identification and classification of woven fabric weave patterns," *Textile Research Journal*, vol. 80, no. 20, pp. 2144-2157, 2010.
- [38] C.F.J. Kuo, C.C. Tsai, "Automatic recognition of fabric nature by using the approach of texture analysis," *Textile research journal*, vol. 76, no. 5, pp. 375-382, 2006.
- [39] A. Conci, C. B. Proença, "A fractal image analysis system for fabric inspection based on box-counting method," *Computer Networks and ISDN Systems*, vol. 30, pp. 1887-1895, 1998.
- [40] H. Sari-Sarraf, J.S. Goddard Jr., "Vision System for On-Loom Fabric Inspection," *IEEE Trans. Industrial Application*, vol. 35, no. 6, pp. 1252-1259, 1999.
- [41] S.W. Zucker, D. Terzopoulos, "Finding structure in co-occurrence matrices for texture analysis," in: A. Rosenfeld (Ed.), *Image Modelling*, Academic Press, New York, pp. 423–445, 1990.
- [42] R.W. Conners, C.W. Mcmillin, K. Lin, R.E. Vasquez-Espinosa, "Identifying and locating surface defects in wood: Part of an automated lumber processing system," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 6, pp. 573-583, 1983.
- [43] J.S. Weszka, C. R. Dyer, A. Rosenfeld, "A comparative study of texture measures for terrain classification," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 4, pp. 269-285, 1976.

- [44] D.A. Clausi, "An analysis of co-occurrence texture statistics as a function of grey level quantization," *Canadian Journal of remote sensing*, vol. 28, no. 1, 45-62, 2002.
- [45] C.F.J. Kuo, T.L. Su, "Gray relational analysis for recognizing fabric defects," *Textile Research Journal*, vol. 73, no. 5, pp. 461-465, 2003.
- [46] L.K. Soh, C. Tsatsoulis, "Texture analysis of SAR sea ice imagery using gray level co-occurrence matrices," *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 37, no. 2, pp. 780-795, 1999.
- [47] W. Gómez, W. C. A. Pereira, A. F. C. Infantosi, "Analysis of Co-Occurrence Texture Statistics as a Function of Gray-Level Quantization for Classifying Breast Ultrasound", *IEEE Trans. Biomed. Eng.*, vol. 31, no. 10, pp. 1889-1899, 2012.
- [48] M.E. Shokr, "Evaluation of second-order texture parameters for sea ice classification from radar images," *Journal of Geophysical Research: Oceans (1978–2012)*, vol. 96, no. C6, pp. 10625-10640, 1991.
- [49] R.F. Walker, P. Jackway, I. D. Longstaff, "Improving co-occurrence matrix feature discrimination," In *DICTA'95, 3rd Conference on Digital Image Computing: Techniques and Application*, pp. 643-648, 1995.
- [50] P. Peng, "Automated defect detection for textile fabrics using Gabor wavelet networks," PhD dissertation, University of Hong Kong, 2006.
- [51] M. Sonka, V. Hlavac, R. Boyle. "Image processing, analysis, and machine vision", Vol. 3, Toronto, Thomson, 2008.
- [52] X. Xie, "A review of recent advances in surface defect detection using texture analysis techniques," *Electronic Letters on Computer Vision and Image Analysis*, vol. 7, no. 3, pp. 1-22, 2008.
- [53] D. Chetverikov, A. Hanbury, "Finding defects in texture using regularity and local orientation," *Pattern Recognition*, vol. 35, no. 10, pp. 2165-2180, 2002.
- [54] E.J. Wood, "Applying Fourier and associated transforms to pattern characterization in textiles," *Textile Research Journal*, vol. 60, no. 4, pp. 212-220, 1990.
- [55] L.H. Siew, R.M. Hodgson, E.J. Wood, "Texture measures for carpet wear assessment," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 10, no. 1, pp. 92-105, 1988.
- [56] S. Sardy, L. Ibrahim, Y. Yasuda, "An application of vision system for the identification and defect detection on woven fabrics by using artificial neural networks," *Proc. Int. Joint Conf. Neural Networks*, vol.3, pp. 2141-2144, 1993.
- [57] M. Singh, S. Singh, "Spatial texture analysis: a comparative study," In *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, vol. 1, pp. 676-679, IEEE, 2002.
- [58] C. Chan, G.K.H. Pang, "Fabric defect detection by Fourier analysis," *Industry Applications, IEEE Transactions on*, vol. 36, no. 5, pp. 1267-1276, 2000.

- [59] C. Castellini, F. Francini, G. Longobardi, B. Tiribilli, P. Sansoni, "On-line textile quality control using optical Fourier transforms," *Optics and lasers in engineering*, vol. 24, no. 1, pp. 19-32, 1996.
- [60] S.A.H. Ravandi, K. Toriumi, "Fourier Transform Analysis of plain Weave Fabric Appearance," *Textile Research Journal*, vol. 65, no. 11, pp. 676-683, 1995.
- [61] D.M. Tsai, C.Y. Hsieh, "Automated surface inspection for directional textures," *Image and Vision Computing*, vol. 18, no. 1, pp. 49-62, 1999.
- [62] P. Kovesi, (2014, Sept. 10). *What are log-Gabor filters and why they are good* [Online]. Available: <http://www.csse.uwa.edu.au/~pk/research/matlabfns/PhaseCongruency/Docs/convexpl.html>
- [63] A.C. Bovik, M. Clark, W.S. Geisler, "Multichannel texture analysis using localized spatial filters," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 12, no. 1, pp. 55-73, 1990.
- [64] D. Dunn, W.E. Higgins, "Optimal Gabor filters for texture segmentation," *Image Processing, IEEE Transactions on*, vol. 4, no. 7, pp. 947-964, 1995.
- [65] A.K. Jain, F. Farrokhnia, "Unsupervised texture segmentation using Gabor filters," In *Systems, Man and Cybernetics, 1990. Conference Proceedings., IEEE International Conference on*, pp. 14-19. IEEE, 1990.
- [66] V. Levesque, "Texture segmentation using Gabor filters," *Center for Intelligent Machines Journal, McGill University*, 2000.
- [67] A. Bodnarova, M. Bennamoun, S. Latham, "Optimal Gabor filters for textile flaw detection," *Pattern recognition*, vol.35, no. 12, pp. 2973-2991, 2002.
- [68] J. Escofet, R. Navarro, J. Pladellorens, "Detection of local defects in textile webs using Gabor filters," *Optical Engineering*, vol. 37, no. 8, pp. 2297-2307, 1998.
- [69] A. Kumar, G.K.H. Pang, "Defect detection in textured materials using Gabor filters," *Industry Applications, IEEE Transactions on*, vol. 38, no. 2, pp. 425-440, 2002.
- [70] K.L. Mak, P. Peng, "An automated inspection system for textile fabrics based on Gabor filters," *Robotics and Computer-Integrated Manufacturing*, vol. 24, no. 3, pp. 359-369, 2008.
- [71] J. Jing, H. Zhang, J. Wang, P. Li, J. Jia, "Fabric defect detection using Gabor filters and defect classification based on LBP and Tamura method," *Journal of the Textile Institute*, vol. 104, no. 1, pp. 18-27, 2013.
- [72] I.W. Selesnick, R.G. Baraniuk, N.C. Kingsbury, "The dual-tree complex wavelet transform," *Signal Processing Magazine, IEEE*, vol. 22, no. 6, pp. 123-151, 2005.
- [73] D.J. Field, "Relations between the statistics of natural images and the response properties of cortical cells," *JOSA A*, vol. 4, no. 12, pp. 2379-2394, 1987.

- [74] M. Wang, J. Li, T. Huang, Y. Tian, L. Duan, G. Jia, "Saliency detection based on 2d log-gabor wavelets and center bias," In *Proceedings of the international conference on Multimedia*, pp. 979-982. ACM, 2010.
- [75] K. Kanagalakshmi, E. Chandra, "Frequency Domain Enhancement algorithm based on Log-Gabor Filter in FFT Domain," *European Journal of Scientific Research*, vol. 74, no. 4, pp. 563-573, 2012.
- [76] H. Mohammad, T. Al Khatib, L. Rajab, "Fabric Defect Detection Using Image Fusion in Log Gabor Filter," *Journal of American Science*, vol. 8, no. 9, 2012.
- [77] M. Popovic, "Texture analysis using 2d wavelet transform: Theory and applications," In *Telecommunications in Modern Satellite, Cable and Broadcasting Services, 1999. 4th International Conference on*, vol. 1, pp. 149-158, IEEE, 1999.
- [78] S. Mallat, "A wavelet tour of signal processing," Academic press, 1999.
- [79] D.M. Tsai, B. Hsiao, "Automatic surface inspection using wavelet reconstruction." *Pattern Recognition*, vol. 34, no. 6, pp. 1285-1305, 2001.
- [80] X. Yang, G. Pang, N. Yung, "Fabric defect detection using adaptive wavelet," In *Acoustics, Speech, and Signal Processing, 2001. Proceedings.(ICASSP'01). 2001 IEEE International Conference on*, vol. 6, pp. 3697-3700, IEEE, 2001.
- [81] X. Yang, G. Pang, N. Yung, "Robust fabric defect detection and classification using multiple adaptive wavelets." *IEE Proceedings-Vision, Image and Signal Processing*, vol. 152, no. 6, pp. 715-723, 2005.
- [82] W. Jasper, J. Joines, J. Brenzovich, "Fabric defect detection using a genetic algorithm tuned wavelet filter," *Journal of the Textile Institute*, vol. 96, no. 1, pp. 43-54, 2005.
- [83] N. Heidari, R. Azmi, B. Pishgoo, "Fabric Textile Defect Detection, by Selecting a Suitable Subset of Wavelet Coefficients, through Genetic Algorithm," *International Journal of Image Processing (IJIP)*, vol. 5, no. 1, 2011.
- [84] M. Ralló, M. S. Millán, J. Escofet, R. Navarro, "Wavelet based techniques for textile inspection," *Opt. Eng.*, vol. 26, no. 2, pp. 838-844, 2003.
- [85] H.Y.T. Ngan, G.K.H. Pang, S. P. Yung, M.K. Ng, "Wavelet based methods on patterned fabric defect detection," *Pattern Recognition*, vol. 38, no. 4, pp. 559-576, 2005.
- [86] Y. Han, P. Shi, "An adaptive level-selecting wavelet transform for texture defect detection," *Image and Vision Computing*, vol. 25, no. 8, pp. 1239-1248, 2007.
- [87] X.Z. Yang, G.K.H. Pang, N.H.C. Yung, "Discriminative fabric defect detection using adaptive wavelets," *Optical Engineering*, vol. 41, pp. 3116, 2002.
- [88] X.Z. Yang, G.K.H. Pang, N.H.C. Yung, "Discriminative training approaches to fabric defect classification based on wavelet transform," *Pattern Recognition*, vol. 37, pp. 889-899, 2004.

- [89] D.A. Karras, S. A. Karkanis, B. G. Mertzios, "Supervised and unsupervised neural network methods applied to textile quality control based on improved wavelet feature extraction techniques," *International journal of computer mathematics*, vol. 67, no. 1-2, pp. 169-181, 1998.
- [90] A. Mojsilovic, D. Rackov, M. Popovic, "On the selection of an optimal wavelet basis for texture characterization," *Image Processing, 1998. ICIP 98. Proceedings. 1998 International Conference on*, vol.3, no., pp. 678-682, 1998.
- [91] N. Ahuja, S. Lertrattanapanich, N. K. Bose, "Properties determining choice of mother wavelet," *IEE Proceedings-Vision, Image and Signal Processing*, vol.152, no. 5, pp. 659-664, 2005.
- [92] A. Laine, J. Fan, "Texture classification by wavelet packet signatures," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 15, no. 11, pp. 1186-1191, 1993.
- [93] S.G. Liu, P.G. Qu, "Inspection of fabric defects based on wavelet analysis and BP neural network," In *Wavelet Analysis and Pattern Recognition, 2008. ICWAPR'08. International Conference on*, vol. 1, pp. 232-236. IEEE, 2008.
- [94] A. Serdaroglu, A. Ertuzun, A. Erçil, "Defect detection in textile fabric images using wavelet transforms and independent component analysis," *Pattern Recognition and Image Analysis*, vol. 16, no. 1, pp. 61-64, 2006.
- [95] C.F.J. Kuo, C.Y. Kao, "Self-organizing map network for automatically recognizing color texture fabric nature," *Fibers and Polymers*, vol. 8, no. 2, pp. 174-180, 2007.
- [96] T.L. Su, F.C. Kung, Y.L. Kuo, "Application of back-propagation Neural Network Fuzzy Clustering in textile texture automatic recognition system," In *Wavelet Analysis and Pattern Recognition, 2008. ICWAPR'08. International Conference on*, vol. 1, pp. 46-49. IEEE, 2008.
- [97] S. Arivazhagan, L. Ganesan, "Texture classification using wavelet transform," *Pattern recognition letters*, vol. 24, no. 9, pp. 1513-1521, 2003.
- [98] L. Wang, Z. Deng, X. Wang, "Application of Wavelet Transform Method for Textile Material Feature Extraction", In: D. Baleanu (ed.) *Wavelet Transforms and Their Recent Applications in Biology and Geoscience*, InTech , pp. 207-224, 2012.
- [99] N.G. Kingsbury, "The dual-tree complex wavelet transform: a new technique for shift invariance and directional filters," *Proc. 8th IEEE DSP Workshop*, vol. 8, Utah, 1998.
- [100] N.G. Kingsbury, "The dual-tree complex wavelet transform: a new efficient tool for image restoration and enhancement," *Proc. EUSIPCO*, vol. 98, 1998.
- [101] M. Costin, A. Ignat, "Pitfalls in using Dual Tree Complex Wavelet Transform for texture featuring: A discussion," *2011 IEEE 7th International Symposium on Intelligent Signal Processing*, no. 1, pp. 1-6, 2011.
- [102] H.Z. Wang, X.H. He, W.J. Zai, "Texture image retrieval using dual-tree complex wavelet transform," In *Wavelet Analysis and Pattern Recognition*,

2007. *ICWAPR'07. International Conference on*, vol. 1, pp. 230-234, IEEE, 2007.

- [103] A. H. Kam, T. T. Ng, N. G. Kingsbury, W. J. Fitzgerald, "Content based image retrieval through object extraction and querying," In *Content-based Access of Image and Video Libraries, 2000. Proceedings. IEEE Workshop on*, pp. 91-95. IEEE, 2000.
- [104] E.H.S. Lo, M.R. Pickering, M.R. Frater, J.F. Arnold, "Image segmentation using invariant texture features from the double dyadic dual-tree complex wavelet transform," In *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, vol. 1, IEEE, pp. 1-609, 2007.
- [105] S. Hatipoglu, S.K. Mitra, N.G. Kingsbury, "Texture classification using dual-tree complex wavelet transform," In *Image Processing And Its Applications, 1999. Seventh International Conference on (Conf. Publ. No. 465)*, vol. 1, pp. 344-347. IET, 1999.
- [106] T. Celik, T. Tjahjadi, "Multiscale texture classification using dual-tree complex wavelet transform," *Pattern Recognition Letters*, vol. 30, no. 3, pp. 331-339, 2009.
- [107] S. Geman, D. Geman, "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 6, pp. 721-741, 1984.
- [108] J.M. Hammersley, P. Clifford, "Markov fields on finite graphs and lattices," 1971. Unpublished.
- [109] S.Z. Li, "Markov random field modeling in image analysis," Third edition, Springer, 2009.
- [110] L. Wang, J. Liu, "Texture classification using multiresolution Markov random field models," *Pattern Recognition Letters*, vol. 20, no. 2 pp. 171-182, 1999.
- [111] R. Chellappa, S. Chatterjee, "Classification of textures using Gaussian Markov random fields," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 33, no. 4, pp. 959-963, 1985.
- [112] R.G. Cross, A.K. Jain, "Markov random field texture models," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol.1, pp. 25-39, 1983.
- [113] F.S. Cohen, Z. Fan, S. Attali, "Automated inspection of textile fabrics using textural models," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 13, no. 8, pp. 803-808, 1991.
- [114] O.S. Al-Kadi, "Combined statistical and model based texture features for improved image classification," In *Advances in Medical, Signal and Information Processing, 2008. MEDSIP 2008. 4th IET International Conference on*, pp. 1-4. IET, 2008.
- [115] D.A. Clausi, B. Yue, "Comparing cooccurrence probabilities and Markov random fields for texture analysis of SAR sea ice imagery," *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 42, no. 1, pp. 215-228, 2004.

- [116] E. Miyamoto, T. Merryman, "Fast calculation of Haralick texture features," [Online] Available: ece.cmu.edu/~pueschel/teaching/18799BCMUspring05/material/eizantad.pdf, 2005.
- [117] M.A. Tahir, A. Bouridane, F. Kurugollu, A. Amira, "Accelerating the computation of GLCM and Haralick texture features on reconfigurable hardware," In *Image Processing, 2004. ICIP'04. 2004 International Conference on*, vol. 5, pp. 2857-2860. IEEE, 2004.
- [118] D.A. Clausi, M. E. Jernigan, "A fast method to determine co-occurrence texture features," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 36, no. 1, pp. 298-300, 1998.
- [119] A.E. Svolos, A. Todd-Pokropek, "Time and space results of dynamic texture feature extraction in MR and CT image analysis," *Information Technology in Biomedicine, IEEE Transactions on*, vol. 2, no. 2, pp. 48-54, 1998.
- [120] D.A. Clausi, Y. Zhao, "Rapid extraction of image texture by co-occurrence using a hybrid data structure," *Computers & geosciences*, vol. 28, no. 6, pp. 763-774, 2002.
- [121] D.A. Clausi, Y. Zhao, "Grey level co-occurrence integrated algorithm (GLCIA): a superior computational method to rapidly determine co-occurrence probability texture features," *Computers & Geosciences*, vol. 29, no. 7, pp. 837-850, 2003.
- [122] M. Unser, "Sum and difference histograms for texture classification," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 1, pp. 118-125, 1986.
- [123] M. Misiti, Y. Misiti, G. Oppenheim, J.M. Poggi, "Wavelet Toolbox™ 4 – Getting Started Guide," *Mathworks®*, 2011.
- [124] Wikipedia (2014, July 30). *Wavelet* [Online]. Available: <http://en.wikipedia.org/wiki/Wavelet>.
- [125] S.G. Mallat, "A theory for multiresolution signal decomposition: the wavelet representation," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 11, no. 7, pp. 674-693, 1989.
- [126] D. Bhonsle, S. Dewangan, "Comparative study of dual-tree complex wavelet transform and double density complex wavelet transform for image denoising using wavelet-domain," *International Journal of Scientific and Research Publications*, vol. 2, no. 7, pp. 1-5, 2012.
- [127] M. Unser, "Texture classification and segmentation using wavelet frames," *Image Processing, IEEE Transactions on*, vol. 4, no. 11, pp. 1549-1560, 1995.
- [128] R. Manthalkar, P.K. Biswas, B.N. Chatterji, "Rotation and scale invariant texture features using discrete wavelet packet transform," *Pattern recognition letters*, vol. 24, no. 14, pp. 2455-2462, 2003.
- [129] M.N. Do, M. Vetterli, "Rotation invariant texture characterization and retrieval using steerable wavelet-domain hidden Markov models," *Multimedia, IEEE Transactions on*, vol. 4, no. 4, pp. 517-527, 2002.

- [130] G. Lambert, F. Bock, "Wavelet methods for texture defect detection," In *Image Processing, 1997. Proceedings., International Conference on*, vol. 3, pp. 201-204, IEEE, 1997.
- [131] C.F.J. Kuo, C.C. Tsai, "Automatic recognition of fabric nature by using the approach of texture analysis," *Textile research journal*, vol. 76, no. 5, pp. 375-382, 2006.
- [132] Y.A. Karayiannis, R. Stojanovic, P.Mitropoulos, C. Koulamas, T. Stouraitis, S. Koubias, G. Papadopoulos, "Defect detection and classification on web textile fabric using multiresolution decomposition and neural networks," In *Electronics, Circuits and Systems, 1999. Proceedings of ICECS'99. The 6th IEEE International Conference on*, vol. 2, pp. 765-768. IEEE, 1999.
- [133] S. Guan, X. Shi, H. Cui, Y. Song, "Fabric defect detection based on wavelet characteristics," In *Computational Intelligence and Industrial Application, 2008. PACIIA'08. Pacific-Asia Workshop on*, vol. 1, pp. 366-370, IEEE, 2008.
- [134] A. Basturk, H. Ketencioglu, Z. Yugnak, M. Emin Yuksel, "Inspection of defects in fabrics using Gabor wavelets and principle component analysis," In *Signal Processing and Its Applications, 2007. ISSPA 2007. 9th International Symposium on*, pp. 1-4. IEEE, 2007.
- [135] G. Piella, M. Campedel, B. Pesquet-Popescu, "Adaptive wavelets for image representation and classification," In *European Signal Processing Conference (invited paper)*, 2005.
- [136] Y.L. Qiao, C.Y. Song, "Double-Density Dual-Tree Wavelet Transform Based Texture Classification," In *Intelligent Information Hiding and Multimedia Signal Processing, 2009. IIH-MSP'09. Fifth International Conference on*, pp. 1322-1325, IEEE, 2009.
- [137] Y.L. Qiao, F. Wang, "Dynamic texture classification based on dual-tree complex wavelet transform," In *Instrumentation, Measurement, Computer, Communication and Control, 2011 First International Conference on*, pp. 823-826, IEEE, 2011.
- [138] M. Petrou, P.G. Sevilla, "Image processing: dealing with texture," Vol. 10, Chichester, Wiley, 2006.
- [139] Y. Zhao, L. Zhang, P. Li, B. Huang, "Classification of high spatial resolution imagery using improved Gaussian Markov random-field-based texture features," *Geoscience and Remote Sensing, IEEE Transactions on*, vol. 45, no. 5, pp. 1458-1468, 2007.
- [140] M. Torres-Torriti, A. Jouan, "Gabor vs. GMRF features for SAR imagery classification," In *Image Processing, 2001. Proceedings. 2001 International Conference on*, vol. 3, pp. 1043-1046, IEEE, 2001.
- [141] R.L. Kashyap, R. Chellappa, "Estimation and choice of neighbors in spatial-interaction models of images," *Information Theory, IEEE Transactions on*, vol. 29, no. 1, pp. 60-72, 1983.

- [142] A. Speis, G. Healey, "An analytical and experimental study of the performance of Markov random fields applied to textured images using small samples," *Image Processing, IEEE Transactions on*, vol. 5, no. 3, pp. 447-458, 1996.
- [143] E.R. Davies, "Computer and machine vision: theory, algorithms, practicalities," Academic Press, 2012.
- [144] M.M. Suarez-Alvarez, D.T. Pham, M.Y. Prostov, Y.I. Prostov, "Statistical approach to normalization of feature vectors and clustering of mixed datasets," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Science* :rspa20110704, 2012.
- [145] S. Aksoy, R.M. Haralick, "Feature normalization and likelihood-based similarity measures for image retrieval," *Pattern Recognition Letters*, vol. 22, no. 5, pp. 563-582, 2001.
- [146] B.H. Juang, S. Katagiri, "Discriminative learning for minimum error classification," *Signal Processing, IEEE Transactions on*, vol. 40, no. 12, pp. 3043-3054, 1992.
- [147] K. Fukunaga, "Introduction to statistical pattern recognition," Academic press, 1990.
- [148] A.R. Webb, K.D. Copsey, "Statistical pattern recognition," John Wiley & Sons, 2011.
- [149] M.H. Beale, M. T. Hagan, H.B. Demuth, "Neural Network Toolbox™: User's Guide," The MathWorks, Inc., 2011.
- [150] L.T. Chi, "Fabric Defect Detection by Wavelet Transform and Neural Network," Master's Thesis, University of Hong Kong, 2004.
- [151] J. Heaton, "Introduction to neural networks with Java," Heaton Research, Inc., 2008.
- [152] M. Aly, "Survey on multiclass classification methods," *Neural Netw*, pp. 1-9, 2005.
- [153] R. Rifkin, A. Klautau, "In defense of one-vs-all classification," *The Journal of Machine Learning Research*, vol. 5, pp. 101-141, 2004.
- [154] T. Hastie, R. Tibshirani, "Classification by pairwise coupling," *The annals of statistics*, vol. 26, no. 2, pp. 451-471, 1998.
- [155] T. G. Dietterich, G. Bakiri, "Solving multiclass learning problems via error correcting output codes," *Journal of Artificial Intelligence Research*, vol. 39, pp. 1-38, 1995.
- [156] E.L. Allwein, R.E. Schapire, Y. Singer, "Reducing multiclass to binary: A unifying approach for margin classifiers," *The Journal of Machine Learning Research*, vol. 1, pp. 113-141, 2001.
- [157] M. Shi, R. Fu, S. Huang, Y. Guo, B. Xu, "A method of fabric defect detection using local contrast deviation," In *Image and Signal Processing, 2009. CISP'09. 2nd International Congress on*, pp. 1-5. IEEE, 2009.

- [158] J.K. Chandra, P.K. Banerjee, A.K. Datta, "Singular value decomposition method for the detection of defects in woven fabric refined by morphological operation," In *Recent Advances in Intelligent Computational Systems (RAICS), 2011 IEEE*, pp. 541-544. IEEE, 2011.
- [159] S. Priya, T.A. Kumar, V. Paul, "A novel approach to fabric defect detection using digital image processing," In *Signal Processing, Communication, Computing and Networking Technologies (ICSCCN), 2011 International Conference on*, pp. 228-232, IEEE, 2011.
- [160] M. Schael, "Texture defect detection using invariant textural features," In *Pattern Recognition*, pp. 17-24. Springer Berlin Heidelberg, 2001.
- [161] M. Schael, H. Burkhardt, "Automatic Detection of Errors on Textures Using Invariant Grey Scale Features and Polynomial Classifiers," In *Texture Analysis in Machine Vision, volume 40 of Machine Perception and Artificial Intelligence*, 1999.
- [162] L. Bissi, G. Baruffa, P. Placidi, E. Ricci, A. Scorzoni, P. Valigi, "Patch based yarn defect detection using Gabor filters," In *Instrumentation and Measurement Technology Conference (I2MTC), 2012 IEEE International*, pp. 240-244, IEEE, 2012.
- [163] S.A. Cuenca, A. Camara, "New texture descriptor for high-speed Web inspection applications," In *Image Processing, 2003. ICIP 2003. Proceedings. 2003 International Conference on*, vol. 3, pp. III-537, IEEE, 2003.
- [164] F. Ibarra Picó, S. Cuenca Asensi, F. García Crespi, J.J. Lorenzo Quintanilla, J.L. Morales Benavente, "A comparative study of texture analysis algorithms in textile inspection applications," 2001.
- [165] A.S. Tolba, H. A. Khan, A. M. Mutawa, S. M. Alsaleem, "Decision fusion for visual inspection of textiles," *Textile Research Journal*, vol. 80, no. 19, pp. 2094-2106, 2010.
- [166] A.S. Tolba, "Neighborhood-preserving cross correlation for automated visual inspection of fine-structured textile fabrics," *Textile Research Journal*, vol. 81, no. 19, pp. 2033-2042, 2011.
- [167] A.S. Tolba, "Fast defect detection in homogeneous flat surface products." *Expert Systems with Applications*, vol. 38, no. 10, pp. 12339-12347, 2011.
- [168] K. Basibuyuk, K. Coban, A. Ertuzun, "Model based defect detection problem: Particle filter approach." In *Communications, Control and Signal Processing, 2008. ISCCSP 2008. 3rd International Symposium on*, pp. 348-351, IEEE, 2008.
- [169] H. Zheng, S. Nahavandi, L. Pan, Y. Xiang, "Genetic learning based texture surface inspection," In *Proceedings of the International Conference on Computational Intelligence, Robotics and Autonomous Systems: CIRAS 2001: 28-30 November 2001, Singapore. CIRAS Organising Committee*, 2003.
- [170] L. Bissi, G. Baruffa, P. Placidi, E. Ricci, A. Scorzoni, P. Valigi, "Automated defect detection in uniform and structured fabrics using Gabor filters and PCA," *Journal of Visual Communication and Image Representation*, vol. 24, no. 7, pp. 838-845, 2013.

- [171] O.G. Sezer, A. Erçil, A. Ertuzun, "Using perceptual relation of regularity and anisotropy in the texture with independent component model for defect detection," *Pattern recognition*, vol. 40, no. 1, pp. 121-133, 2007.
- [172] S.G. Kim, T.J. Kang, "Texture classification and segmentation using wavelet packet frame and Gaussian mixture model," *Pattern Recognition*, vol. 40, no. 4, pp. 1207-1221, 2007.
- [173] Institute for textile machinery and textile industry of the ETH Zurich, "Katalog der Gewebefehlerarten im Rohgewebe", Second revised edition, September 1989.
- [174] F. Truchetet, O. Laligant, "A review on industrial applications of wavelet and multiresolution based signal-image processing," In *Eighth International Conference on Quality Control by Artificial Vision*, pp. 63560H-63560H, International Society for Optics and Photonics, 2007.
- [175] Z. Cvetkovic, M. Vetterl, "Oversampled filter banks," *Signal Processing, IEEE Transactions on*, vol. 46, no. 5, pp. 1245-1255, 1998.
- [176] Z. Cvetkovic, M. Vetterli, "Discrete-time wavelet extrema representation: design and consistent reconstruction," *Signal Processing, IEEE Transactions on*, vol. 43, no. 3, pp. 681-693, 1995.
- [177] D.B.H. Tay, M. Palaniswami, "Design of approximate Hilbert transform pair of wavelets with exact symmetry," In *Acoustics, Speech, and Signal Processing, 2004. Proceedings.(ICASSP'04). IEEE International Conference on*, vol. 2, pp. ii-921, IEEE, 2004.
- [178] R. Yu, Runyi, H. Ozkaramanli, "Hilbert transform pairs of biorthogonal wavelet bases," *Signal Processing, IEEE Transactions on*, vol. 54, no. 6, pp. 2119-2125, 2006.
- [179] N.G. Kingsbury, "A dual-tree complex wavelet transform with improved orthogonality and symmetry properties," In *Image Processing, 2000. Proceedings. 2000 International Conference on*, vol. 2, pp. 375-378. IEEE, 2000.
- [180] N.G. Kingsbury, "Design of q-shift complex wavelets for image processing using frequency domain energy minimization," In *Image Processing, 2003. ICIP 2003. Proceedings. 2003 International Conference on*, vol. 1, pp. I-1013. IEEE, 2003.
- [181] I.W. Selesnick, "Hilbert transform pairs of wavelet bases," *IEEE Signal Processing Lett.*, vol. 8, no. 6, pp. 170-173, 2001.
- [182] N. G. Kingbury. (2015, Sept. 10). *Matlab code for Q-shift complex wavelet design*, [Online]. Available: <http://www-sigproc.eng.cam.ac.uk/foswiki/pub/Main/NGK/qshiftgen.zip>
- [183] C. Shihua, L. Keyong. (2015, Sept. 10). *Matlab Implementation of Wavelet Transforms*. Electrical Engineering, Polytechnic University, NY. [Online]. Available: <http://eeweb.poly.edu/iselesni/WaveletSoftware/allcode.zip>.
- [184] S.A. Barker, "Image Segmentation Using Markov Random Field Models," Ph.D. Dissertation, University of Cambridge, 1998.

- [185] J.O. Rawlings, S.G. Pantula, D.A. Dickey, "Applied regression analysis: a research tool," 2nd Edition, Springer, 1998.
- [186] I. Jolliffe, "Principal component analysis," 2nd Edition, Springer, 2002.
- [187] D.A. Clausi, "Comparison and fusion of co-occurrence, Gabor and MRF texture features for classification of SAR sea-ice imagery," *Atmosphere-Ocean*, vol. 39, no. 3, pp. 183-194, 2001.
- [188] L.I. Millett, S.H. Fuller, eds. "The Future of Computing Performance: Game Over or Next Level?" National Academies Press, 2011.
- [189] I.W. Selesnick, "The double-density dual-tree DWT," *Signal Processing, IEEE Transactions on*, vol. 52, no. 5, pp. 1304-1314, 2004.