

Washboarding of Corrugated Cardboard

Sven Dieter Wendler
B. App. Sci.

RMIT

Department of Applied Physics
Faculty of Applied Science
Royal Melbourne Institute of Technology
Melbourne, Australia

March 2006

A Thesis Submitted for the Degree of
Doctor of Philosophy

Declaration

I certify that except where due acknowledgement has been made, the work is that of the author alone; the work has not been submitted previously, in whole or in part, to qualify for any other academic award; the content of the thesis is the result of work which has been carried out since the official commencement date of the approved research program; and, any editorial work, paid or unpaid, carried out by a third party is acknowledged.

Sven Dieter Wendler

Date

Acknowledgements

I would like to thank the following people, without whom, this work would not have been possible:

Dr Michael Reich, my principle academic supervisor, Senior Lecturer, Department of Applied Physics, RMIT, for all of his help and good cheer for the duration of this work.

Mr Russell Allan, my secondary supervisor, for his good humour, suggestions and his support at Amcor Research and Technology to make this work possible.

Kathryn Marsh, for helping with the editing of the numerous changes in section, table and figure numbers of the thesis.

And finally, but certainly not least, my parents for their support and encouragement for the whole duration of this work.

Table of Contents

Declaration	i
Acknowledgements	ii
Table of Contents	iii
List of Figures	vii
List of Tables	xi
Glossary	xii
Abbreviations	xii
Selected Symbols	xiii
Summary	xv
Chapter 1 – Introduction	1
1.1 Corrugated cardboard manufacturing process	1
1.1.1 Flute types.....	2
1.2 Washboarding of corrugated cardboard	4
1.2.1 Creation of washboarding	4
1.2.2 Washboarding and its relationship to strain.....	5
1.3 Measurement of washboarding using digital imaging profilometry.....	5
1.4 Finite element analysis.....	6
Chapter 2 – Literature Review	7
2.1 Pulping, paper furnish and strength	7
2.2 Thermal, Moisture and Mechanical Properties of Paper.	10
2.2.1 The effect of relative humidity and temperature upon the moisture content of paper.....	10
2.2.2 Moisture and hygroexpansion	12
2.2.3 The Measurement of Paper Thickness.....	13
2.2.4 Mechanosorptive (accelerated) creep of paper.....	14
2.2.5 Anisotropy of paper	16
2.2.6 Sonic modulus, anisotropy of paper and relationship to elastic modulus... 18	
2.3 The Effect of Moisture upon the Properties of Starch	22
2.4 Washboarding	25
2.4.1 Washboarding and the printing quality of corrugated cardboard.....	28
2.4.2 Quantifying print quality	31
2.4.3 Finite element analysis (FEA).....	32
Chapter 3 –Theory	34
3.1 An overview of Fourier analysis.	34
3.1.1 Fourier series expansion	34
3.1.2 Real and imaginary terms.....	35
3.1.3 Multiplication rule.....	35
3.1.4 Phase determination of a specific frequency.....	35
3.1.5 Discrete Fourier Transform (DFT)	36
3.1.6 Fast Fourier Transform (FFT)	36
3.2 The effect of relative humidity upon wheat starch (glue).....	38
3.2.1 Moisture content of starch.....	38

3.2.2	<i>The effect of moisture upon the elastic modulus of starch</i>	39
3.3	<i>Calculation of the average strain from washboarding profiles</i>	41
3.4	<i>Finite element analysis (FEA) of washboarding</i>	43
3.4.1	<i>Finite element analysis overview</i>	43
3.4.2	<i>Emulation of glue quantity in finite element analysis of washboarding</i>	45
3.5	<i>Measurement of full-tone print coverage</i>	48
Chapter 4 – Methods Part A – Measuring Washboarding		52
4.1	<i>Digital image profilometry - Overview</i>	52
4.1.1	<i>Washboarding Hardware</i>	52
4.1.1(a)	<i>Image acquisition board</i>	53
4.1.1(b)	<i>Charged coupled device (CCD) camera</i>	54
4.1.1(c)	<i>Pneumatic sample holders</i>	54
4.1.2	<i>Washboarding software</i>	54
4.1.2(a)	<i>Extraction of washboarding profile software</i>	54
4.1.2(b)	<i>Extraction of washboarding shape and strain software</i>	55
4.2	<i>Measurement of Washboarding Profiles</i>	56
4.2.1	<i>Projected lines</i>	56
4.2.2	<i>The method of extracting a profile</i>	57
4.2.3	<i>Phase unwrapping</i>	61
4.2.4	<i>Filtering</i>	62
4.2.5	<i>Calibration</i>	63
4.2.6	<i>Final washboarding profile</i>	66
4.3	<i>Original contributions to the profilometer technique</i>	68
Chapter 5 – Methods Part B – Investigations		70
5.1	<i>The effect of starch (glue) quantity upon washboarding</i>	70
5.1.1	<i>Washboard depth and glue coverage – Manually manufactured board</i>	70
5.1.2	<i>Washboard depth and glue coverage – Pilot corrugator manufactured</i>	72
5.1.3	<i>Emulating washboarding using finite element analysis</i>	74
5.2	<i>Mechanical properties of paper and its effect upon washboarding</i>	77
5.2.1	<i>Conditioning of paper samples prior to testing</i>	77
5.2.2	<i>The relationship between the washboarding and the sonic modulus of paper</i>	77
5.3	<i>Relative humidity and its effect upon washboarding</i>	80
5.3.1	<i>The cycling of relative humidity and its effect upon washboarding</i>	80
5.3.2	<i>Washboarding depth</i>	81
5.3.3	<i>Washboarding shape</i>	81
5.4	<i>The effect of washboarding upon the performance of corrugated cardboard</i>	83
5.4.1	<i>Edge crush test performance</i>	83
5.4.1(a)	<i>Empirical Testing</i>	83
5.4.1(b)	<i>Finite element analysis of edge crush test</i>	84
5.4.2	<i>MD-Shear performance</i>	85
5.4.2(a)	<i>Empirical testing</i>	86
5.4.2(b)	<i>Finite element analysis of MD-Shear</i>	87
5.4.3	<i>Three-point bend performance</i>	89
5.4.3(a)	<i>Empirical testing</i>	90
5.4.3(b)	<i>Finite element analysis of three-point bend</i>	90
5.5	<i>The effect of washboarding upon the printing quality of corrugated cardboard</i>	92
5.5.1	<i>Full-tone print</i>	92
5.5.2	<i>Flexographic printer settings</i>	92
5.5.3	<i>Sample selection</i>	93

Chapter 6 - Results and Discussion	96
6.1 Manufacturing process of corrugated cardboard and its relationship to washboarding geometry.....	96
6.1.1 Washboard depth and glue quantity.....	96
6.1.1(a) Manual glue application.....	96
6.1.1(b) Corrugator glue application.....	97
6.1.1(c) Machine speed.....	99
6.2 Mechanical properties of paper and its effect upon washboarding.....	101
6.2.1 Washboarding and its relationship to the sonic modulus of paper.....	101
6.2.2 Washboarding and its relationship to strain.....	103
6.2.3 Finite element analysis of washboarding.....	106
6.2.3(a) Elastic modulus of paper and washboarding.....	109
6.2.3(b) Simulated glue force.....	111
6.3 Environmental conditions and its effect upon washboarding geometry.....	113
6.3.1 The cycling of relative humidity and its effect upon washboarding.....	113
6.3.1(a) Washboarding depth and creep.....	113
6.3.1(b) Washboarding shape and moisture content.....	114
6.3.2 Finite element analysis (FEA) study of moisture in corrugated board.....	116
6.4. The effect of washboarding upon the performance of corrugated cardboard.....	118
6.4.1 Edge crush test (ECT) performance.....	118
6.4.1(a) Empirical testing.....	118
6.4.1(b) Finite element analysis of the edge crush test.....	120
6.4.2 MD-Shear performance.....	125
6.4.2(a) Empirical testing.....	125
6.4.2(b) Finite element analysis of MD-Shear.....	126
6.4.3 Three-point bend test performance.....	128
6.4.3(a) Empirical testing.....	128
6.4.3(b) Finite element analysis of three-point bend testing.....	130
6.4.4 Final box strength and performance measures.....	134
6.5 The effect of washboarding upon the printing quality of corrugated cardboard ...	136
6.5.1 Full-tone coverage.....	136
6.5.2 Humidity, printability and structural integrity.....	138
 Chapter 7 - Conclusions	 139
Suggestions for further work.....	143
A study of the furnish of paper and its effect upon on washboarding for constant and cyclic humidity.....	143
A study into starch additives and its effect upon washboarding.....	143
A study into washboarding and its effect upon box compression strength and survival.....	143
A study on washboarding variability.....	144
A study into how printing quality could be improved by increasing the moisture content of the corrugated cardboard.....	144
 References	 145
 Appendix A -Washboarding profiling program	 A.1
Source code	A.3

<i>Appendix B - Phase and strain program</i>	<i>B.1</i>
<i>Source code</i>	<i>B.2</i>
<i>Appendix C – Paper properties used for finite element analysis</i>	<i>C.1</i>

List of Figures

Figure [1.1.1] – Single facer section of corrugator reproduced from Wright et al. [90].....	1
Figure [1.1.2] – Single facer board reproduced from Wright et al. [90]	2
Figure [1.1.3] – Corrugator reproduced from Wright et al. [90]	2
Figure [1.2.1] – Washboard deformation. Image produced by the author.....	4
Figure [2.2.1] – How moisture content of kraft-liner board varies with relative humidity, reproduced from Benson [8].	10
Figure [2.2.2] – Moisture content of softwood kraft-liner board in constant humidity in an environment of changing temperature, reproduced from Benson [ibid].	11
Figure [2.2.3] – Anisotropic stress-strain relationships of paper in different relative humidities, reproduced from Benson [ibid].	17
Figure [2.2.4] – Polar plot of the speed of sound squared for a typical machine manufactured paper measured by the author using a Nomura Shoji sonic sheet tester at 50 % RH and 23 °C.	19
Figure [2.3.1] – Change in elastic modulus of various wheat starch - polymer combinations for different moisture contents, reproduced from Kirby [41]. Composition: (●) wheat starch; (◆) wheat starch-xylitol; (■) wheat starch-glycerol	23
Figure [2.4.1] – Full tone printing – severe striping (printed at ISO conditions by the author)	29
Figure [2.4.2] – Full tone printing – partial striping (printed at ISO conditions by the author).	29
Figure [2.4.3] – Full tone printing – negligible striping (printed at ISO conditions by the author).	29
Figure [2.4.4] – Crushing of corrugated cardboard fluting. Images produced by the author.	30
Figure [2.4.5] – From Netz [52] showing a decrease on washboarding for increasing grammage	32
Figure [3.2.1] – Relationship between relative humidity of an environment and starch moisture content.	39
Figure [3.2.2] – Relationship between relative humidity of an environment and modulus of elasticity of starch.....	40
Figure [3.3.1] – Geometry of strain measurement from a washboarding profile.	42
Figure [3.5.1] – The area, A_{jk} , used for print coverage analysis.	48
Figure [3.5.2] – The image in figure [3.5.1] increased in contrast for improved visual inspection (not used in analysis).	48
Figure [3.5.3] – Intensity histogram of the values in the area, A_{jk} , as illustrated in image [3.5.1]. It shows the values B and D visually.....	49

<i>Figure [3.5.4] – The area, A_{jk}, after thresh-holding</i>	50
<i>Figure [4.1.1] – Washboard measurement set-up</i>	53
<i>Figure [4.2.1] – An image captured for the purpose of measurement of washboarding. It represents an area of approximately 8 x 8 cm (64 cm²)</i>	56
<i>Figure [4.2.2] – Flow chart describing the method of profile extraction from an image of projected lines.</i>	58
<i>Figure [4.2.3] – The frequency spectrum (left) is constructed from the Fourier Transformation of projected fringe lines (right).</i>	59
<i>Figure [4.2.4] – A typical frequency spectrum across a horizontal line of an image</i>	60
<i>Figure [4.2.5] – Profile before phase unwrapping</i>	61
<i>Figure [4.2.6] – Phase unwrapped profile</i>	62
<i>Figure [4.2.7] – Frequency response of recursive digital filter</i>	62
<i>Figure [4.2.8] – Surface profile after filtering</i>	63
<i>Figure [4.2.9] – Profilometer with calibration steel plate in place.</i>	64
<i>Figure [4.2.10] – Acquired image of steel plate used for calibration.</i>	64
<i>Figure [4.2.11] – Resultant phase profile of image in figure [4.1.11]</i>	65
<i>Figure [4.2.12] – Washboard depth in units of distance and pixels.</i>	65
<i>Figure [4.2.13] – Image for abscissa calibration</i>	66
<i>Figure [4.2.14] – Two profiles showing different severities of washboarding.</i>	67
<i>Figure [5.1.1] – Method of controlling glue application</i>	71
<i>Figure [5.1.2] – Cross section of the washboarding and edge crush model (see figure [5.4.1]) displaying the locations of the simulated glue force by the application of pressure on elements.</i>	75
<i>Figure [5.3.1] – Graph of programmed relative humidity cycle of environmental room.</i>	81
<i>Figure [5.4.1] – Finite element analysis model for washboarding and edge crush test modeling</i>	85
<i>Figure [5.4.2] – Finite element analysis model for modelling MD-Shear.</i>	88
<i>Figure [5.4.3] – A material undergoing a three-point test.</i>	89
<i>Figure [5.4.4] – Finite element analysis model for modelling three-point bend.</i>	91
<i>Figure [6.1.1] – Relationship between glue quantity and washboarding depth for manually glued liners.</i>	97
<i>Figure [6.1.2] – Relationship between glue quantity and washboard depth for corrugator manufactured board.</i>	98
<i>Figure [6.1.3] – Relationship between glue quantity and washboard depth for corrugator manufactured board and manually glued liners.</i>	99

<i>Figure [6.1.4] – Relationships between corrugator glue gap, speed and applied glue quantity.</i>	100
<i>Figure [6.2.1] – Increasing washboarding depth as sonic modulus decreases.</i>	102
<i>Figure [6.2.2] – Relationship between sonic modulus of the paper and washboarding depth.</i>	102
<i>Figure [6.2.3] – Relative frequency components calculated using Fourier transformations of a range of washboarding profiles with different depths, where the depth for the blue curve is 60 % greater than for the red curve.</i>	104
<i>Figure [6.2.4] – Normalised Fourier transformations of a range of washboarding profiles with different depths, where the depth for the blue curve is 60 % greater than for the red curve.</i>	104
<i>Figure [6.2.5] – A typical washboarding profile used for strain calculation</i>	105
<i>Figure [6.2.6] – Relationship between washboard depth and liner strain.</i>	105
<i>Figure [6.2.7] – Relationship between glue quantity and washboarding liner strain.</i>	106
<i>Figure [6.2.8] – Finite element model showing washboarding due to simulated glue force application.</i>	107
<i>Figure [6.2.9] – Finite element model showing stress concentration and washboarding due to simulated glue force application.</i>	108
<i>Figure [6.2.10] – A side on view of the finite element model depicting washboarding.</i>	108
<i>Figure [6.2.11] – An exaggerated view of washboarding and fluting deformation (10 x normal displacement) illustrating a reduction in corrugated cardboard thickness.</i>	109
<i>Figure [6.2.12] – Relationship between paper modulus and washboarding depth using finite element analysis.</i>	110
<i>Figure [6.2.13] – Relationship between glue pressure and washboarding depth using finite element analysis.</i>	111
<i>Figure [6.3.1] – Change in washboarding depth over time due to the cycling of relative humidity.</i>	113
<i>Figure [6.3.2] – The relationships between the magnitude of the first harmonic, cyclic relative humidity and washboarding depth.</i>	114
<i>Figure [6.3.3] – The geometry of washboarding after five humidity cycles (day 5) for the sample at 50, 65, and 90 % relative humidities.</i>	115
<i>Figure [6.3.4] – Shape of washboarding sampled at the first cycle (day 1) and at the sixth cycle (day 6) of relative humidity.</i>	115
<i>Figure [6.3.5] – Relationship between relative humidity and washboard depths using finite element analysis.</i>	117
<i>Figure [6.4.1] – Relationship between washboarding depth and ECT failure load.</i>	118
<i>Figure [6.4.2] – Relationship between total board grammage and ECT failure load.</i>	119
<i>Figure [6.4.3] – Relationship between board grammage and washboarding depth.</i>	120
<i>Figure [6.4.4] – A y-displacement colour map of a finite element model depicting an ECT sample under load</i>	121
<i>Figure [6.4.5] – A stress colour map of a finite element model depicting an ECT sample under load.</i>	121

<i>Figure [6.4.6] – Mapping of washboard depth, ECT failure load, percentage simulated glue force and paper properties.</i>	123
<i>Figure [6.4.7] – Relationship between simulated glue force, paper elastic modulus and ECT failure load.</i>	124
<i>Figure [6.4.8] – Relationship between washboarding depth and MD-Shear values.</i>	125
<i>Figure [6.4.9] – A y-displacement colour map of a finite element model of a MD-Shear sample undergoing testing.</i>	127
<i>Figure [6.4.10] – MD-Shear load versus rotation graphs for a range of washboarding depths using finite element analysis.</i>	127
<i>Figure [6.4.11] – Relationship between washboarding depth and MD-Shear values.</i>	128
<i>Figure [6.4.12] – Relationship between washboarding depth and load at failure.</i>	130
<i>Figure [6.4.13] – A colour stress map of a three-point bend model after the application of simulated glue force.</i>	131
<i>Figure [6.4.14] – Side on view of finite element model depicting a three-point test at failure.</i>	131
<i>Figure [6.4.15] – Finite element model depicting a y-displacement colour map of a three-point test at failure.</i>	132
<i>Figure [6.4.16] – Finite element model depicting a stress colour map of a three-point test at failure.</i>	132
<i>Figure [6.4.17] – Load versus displacement graphs for a range of glue forces (and therefore washboarding) calculated using finite element analysis.</i>	133
<i>Figure [6.4.18] – Relationship between washboarding depth and three-point bend failure.</i>	134
<i>Figure [6.5.1] – Percentage area of no print coverage for a given washboarding depth and constant printing pressure on bleached white liners.</i>	136
<i>Figure [6.5.2] – Percentage area of no print coverage for a given washboarding depth and constant printing pressure on unbleached brown liners. Samples with measurement of print ratio greater than 96 % (less than 4 % 100-PR) appear to have no striping as judged by a human eye (see the area defined).</i>	137
<i>Figure [7.1] – The effect of washboarding upon corrugated performance measures. *For washboarding depths less than 40 μm.</i>	141

List of Tables

<i>Table [1.1.1] – Geometry of fluting. *Pitch refers to the wavelength of the fluting, ie. the distance from tip to tip.....</i>	<i>3</i>
<i>Table [3.2.1] – Moisture content and modulus of elasticity of starch for a range of relative humidities of the environment controlled by saturated salt solutions [58].....</i>	<i>38</i>
<i>Table [3.4.1] – Modulus of elasticity of glue for a range of relative humidities of the environment and relative simulated glue force applied in finite element analysis. *Simulated glue force is relative to the force applied by the starch at 50% relative humidity.....</i>	<i>45</i>
<i>Table [5.1.1] – Basis weight and compositions of papers used in glue quantity investigation - Manually manufactured.....</i>	<i>70</i>
<i>Table [5.1.2] – Basis weight and compositions of papers used in glue quantity investigation - Pilot corrugator manufactured.....</i>	<i>72</i>
<i>Table [5.2.1] – Basis weight and compositions of papers used in washboarding depth versus elastic modulus investigation.</i>	<i>78</i>
<i>Table [5.3.1] – Basis weight and compositions of papers used in cyclic humidity investigation.</i>	<i>80</i>
<i>Table [5.4.1] – Basis weight and compositions of papers used in the MD-Shear and three-point bend investigations.....</i>	<i>87</i>
<i>Table [5.5.1] – Washboarding samples used for the full-tone printability study.....</i>	<i>94</i>
<i>Table [6.1.1] – Results of manual glue study.</i>	<i>96</i>
<i>Table [6.1.2] – Results of the corrugator glue study.....</i>	<i>97</i>
<i>Table [6.2.1] – Results of the liner orientation study.....</i>	<i>101</i>
<i>Table [6.3.1] – Summary of paper properties, glue properties and washboarding results for four different relative humidities using finite element analysis.....</i>	<i>116</i>
<i>Table [6.4.1] – Results of edge crush test (ECT) study. *Standard deviation of eight repetitions.</i>	<i>118</i>
<i>Table [6.4.2] – Parameters used in the sixteen finite element models to investigate ECT. Also shown are the resulting ECT failure loads and washboarding depths produced.</i>	<i>122</i>
<i>Table [6.4.3] – Results for MD-Shear study. *Standard deviation of eight repetitions.....</i>	<i>125</i>
<i>Table [6.4.4] – Data of three-point test study. *Standard deviation of eight repetitions</i>	<i>129</i>
<i>Table [6.5.1] – Results of washboard depth and print coverage study.....</i>	<i>136</i>

Glossary

Abbreviations

AS	Australian Standard
CD	Cross Direction
CCD	Charged Coupled Device
DFT	Discrete Fourier Transform
DFFT	Discrete Fast Fourier Transform
DLL	Dynamic Link Library
DOF	Degrees of Freedom
ECT	Edge Crush Test
FEA	Finite Element Analysis
FFT	Fast Fourier Transform
FS	Strong Fluting
FT	Fourier Transform
ISO	International Standards Organization
KL	Kraft Liner
MD	Machine Direction
NSSC	Neutral Sulphite Semi-Chemical
PR	Print Ratio
RH	Relative Humidity
TK	Top Kraft
TSO	Tensile Stiffness Orientation
WL	White Liner
XRF	X-Ray Fluorescence

Selected Symbols

ε	Strain
ϕ	Angle of twist for an MD-shear sample
ρ	Density (kg/m ³)
τ	Torque (Nm)
ω	Angular Frequency (s ⁻¹)
A_b	Area of board (m ²)
A_g	Cross sectional area of glue (m ²)
A_{jk}	Area under investigation in print coverage
B_I	Mean of the values in A_{jk}
B_w	Final board weight (g)
D_{3pp}	Displacement of three-point bend sample
D_P	Mean of the values in P_{jk}
D_{wb}	Washboarding depth of a single point (m)
E	Modulus of Elasticity (Pa)
E_s	Sonic modulus (Pa)
F	Force (N)
G_c	Glue coverage (g/m ²)
G_w	Glue weight (g)
I	Second moment of area (m ⁴)
I_{jk}	Area defining board intensity levels
k	Spring constant (N/m)
L_{MB}	Length of an MD-shear sample
L_{pp}	Distance between three-point bend supports
L_w	Liner weight (g)
L_{wb}	Length between washboarding peaks (m)

M	MD-shear value
P_{jk}	Area defining ink intensity levels
PR	Print ratio (%)
Sf_w	Single facer weight (g)
t	Time (s)
T	Length of a periodic signal (m)
T_A	Mean of the Values A_{jk}
T_{FFT}	Time taken for a Fourier transform (s)

Summary

The aim of this thesis was to study the causes of washboarding and its impact upon the structural integrity and printability of corrugated cardboard packaging.

The term washboarding refers to the undulating surface of a corrugated board formed by the shrinkage of glue in between the liner and the fluting of corrugated board.

A digital image profilometry technique was developed to measure two-dimensional washboarding profiles of corrugated board. This instrument, in conjunction with the software package written for this application, was found to be reliable, robust and sufficiently quick to allow a range of studies of the effects of washboarding that were not previously feasible. The profilometer was used to measure the washboarding depth and washboarding profiles for a range of corrugated boards, some assembled manually and some machine manufactured. As there has been little published information on the profiles of washboarding due to the difficulties in acquiring such measurements, much of the studies presented in this thesis are novel and can also be the basis for more extensive studies.

The mechanical properties of paper and its effect upon washboarding depth were studied. These included the relationship between the degree of washboarding and the sonic modulus of the paper and the resultant strain in the liner. Finite Element Analysis (FEA) was used to model washboarding. This modelling was used to confirm the experimental results and provided insights into the mechanisms responsible for washboarding. The effect that corrugator machine speed had upon the degree of washboarding was also determined.

The primary conclusions were as follows:

- Washboarding depth was linearly related to the amount of glue applied.
- A change in corrugator manufacturing rate increased the amount of glue applied, thus increasing the amount of washboarding. When the glue quantity was similar for different machine speeds there was little to no change to washboarding depth, suggesting that machine speed by itself does not necessarily change the depth of washboarding
- A linear change in washboarding depth resulted in a non-linear change in strain. The strain displayed a power law dependence upon washboarding.
- Washboarding depth increased non-linearly with a decrease in the measured sonic modulus. A large increase in washboarding was seen where the measured sonic modulus of the board liner was less than 5.0 GPa. The finite element modelling of the effect of a change in paper modulus upon washboarding also displayed a sharp rise as the elastic modulus decreased. A paper with a typical glue quantity applied was modelled, for which the washboarding depth was seen to greatly increase when the Young's modulus of the paper was less than 5.5 GPa.

The effect of environmental conditions upon washboarding geometry was studied. An environmental room was used to cycle relative humidity, during which the profilometer system was used to measure the washboarding profile of corrugated board at specific times over a period of five days. Changing the relative humidity had the effect of changing the moisture content of paper and glue and therefore its mechanical properties. These profiles were analysed for washboarding depth and washboarding shape. Fourier analysis was used to describe the shape of the washboarding as humidity and therefore the moisture content of the corrugated board changed.

From the environmental studies it was found that:

- Washboarding depth was highly dependent on the relative humidity of the environment, where an increase in relative humidity decreased the washboarding depth in a linear fashion
- The shape of washboarding changed with relative humidity, but the change was minor when compared to the change in absolute washboarding depth
- As the relative humidity increased, the washboarding depth decreased linearly. It was known previously that as moisture content increased (due to increasing relative humidity) the elastic modulus of paper and glue decreased, with an accompanying increase in the thickness of the paper. It was found that the reduction in paper stiffness effectively negated the increase in bending stiffness. This left the change in glue force as having had the greatest effect upon washboarding depth and the primary mechanism that altered the washboarding depth when the relative humidity changed.

The effects of washboarding upon three performance measures were tested empirically and modelled using Finite Element Analysis. The performance measures studied were edge crush test (ECT), three-point bend, and MD-Shear (an Amcor Ltd. proprietary test). These performance measures give an indication of how well the corrugated board may perform in a box under load, such as it would experience at the bottom of a pallet.

These studies gave the following findings:

- Edge crush test (ECT) performance was found to be strongly correlated with both washboarding depth and paper grammage. FEA was used to clarify these relationships. It was found that both the geometry of the washboarding due to changing the simulated glue force and the changing stiffness of the paper affected the resultant ECT performance. In general, it was found that an increase in glue quantity decreased ECT performance while

an increase in the elastic modulus of the paper (as a result of an increase in grammage) increased ECT performance.

- The use of three-point bend testing found that a doubling in washboarding depth increased performance for this measurement by only 3 %. FEA supported this result.
- MD-shear measurements showed an increase in performance for this measure as washboarding depth increased. The increase in performance observed in physical testing (7 %) was roughly double the increase (4 %) seen using FEA when washboarding depth increased from 25 μm to 40 μm . This was possibly due to the differences in samples size, where the physical size of an MD-Shear test piece was larger than that modelled in FEA due to computational limitations.

Finally, the effect of washboarding upon the printing quality of corrugated board was studied. A method was developed to measure full-tone print coverage of corrugated board. Image analysis was used to determine the full-tone print coverage, applied using a flexographic printer. A range of samples was measured for washboarding depth and full tone coverage.

It was found that an increase in washboarding decreased the full-tone print coverage (ie. a reduction of print quality) and if the washboarding depth was kept sufficiently low, then full coverage would occur for a given printing pressure. It was also found that a linear relationship existed between washboarding depth and coverage for a single printing pressure. Two separate relationships were found, one for bleached liner and another for unbleached liner.

Chapter 1 – Introduction

Washboarding is the term used in the paper packaging industry to describe the unwanted undulations on the surface of corrugated cardboard.

1.1 Corrugated cardboard manufacturing process

Corrugated cardboard is fabricated from four basic components. These are face-liner, back-liner, medium and starch (glue). A corrugated cardboard piece is manufactured by feeding a hot steamed (and therefore wet) medium into a corrugator (figure [1.1.1]). Two very hot (> 170 °C) interleaved corrugated rolls form the fluting from the medium. The fluting is then glued to the face-liner in one continuous process. The product created (shown in figure [1.1.2]) is called single facer.

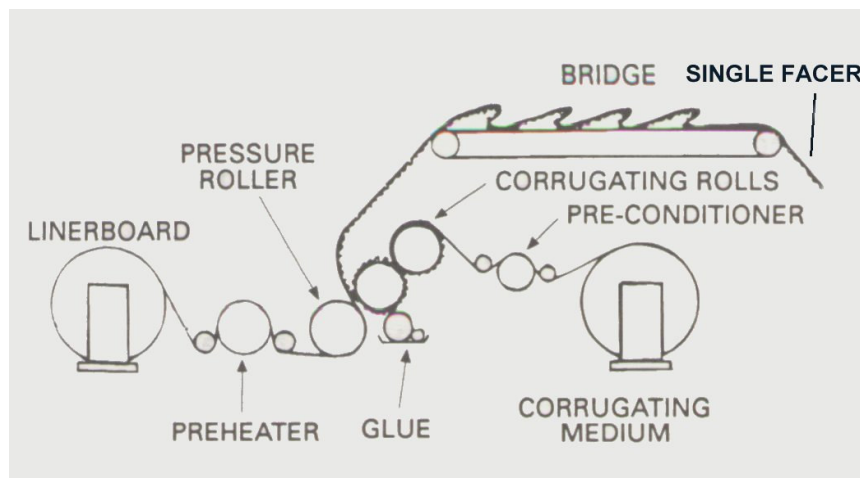


Figure [1.1.1] – Single facer section of corrugator reproduced from Wright et al. [1]

To form the final corrugated board, the single facer is fed into the double backer section of the machine (shown in figure [1.1.3]) which applies starch to the flutes and then glues a second liner to it. The second liner is also known as the double backer.



Figure [1.1.2] – Single facer board reproduced from Wright et al. [1]

Numerous parameters can be altered to suit the specific applications of the final product. Some of the variables that may be altered include, but are not limited to, different fibre content (furnish), basis weights, stiffness and/or paper thickness. In addition, the glue gap setting can be altered, varying the quantity of starch applied. Different types of starch and starch concentrations may also be used. The geometry and size of the flute can be changed by the use of different corrugator profiles (see section [1.1.1] below).

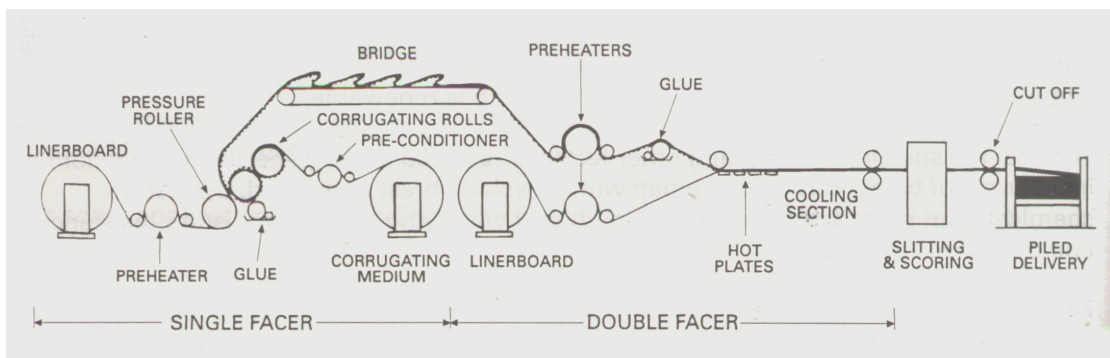


Figure [1.1.3] – Corrugator reproduced from Wright et al. [1]

1.1.1 Flute types

Flute types come in a range of sizes and geometry. The most common types are shown in table [1.1.1], where the two most frequently used types are B-flute and C-flute. Take-up factor relates to the quantity of flute used per linear length of board, e.g. C-flute with a take up factor of 1.48 needs 48 % more paper for any given length of corrugated cardboard, in comparison to the liner.

Table [1.1.1] – Geometry of fluting. *Pitch refers to the wavelength of the fluting, ie. the distance from tip to tip.

Flute Type	Pitch* (mm)	Height (mm)	Typical take-up factor
A	8.0 - 9.0	4.6 - 4.8	1.56
C	7.0 - 8.0	3.2 - 3.8	1.48
B	6.0 - 6.5	2.4 - 2.7	1.37
E	3.0 - 3.6	1.0 - 1.8	1.28
F	~1.5	~0.7	~1.2

This variability in manufacturing is common as the final box product may be used for different purposes due to economics and/or environmental factors, such as the use of recycled fibres in the furnish. All of these parameters may alter washboarding and/or final box performance.

1.2 Washboarding of corrugated cardboard

1.2.1 Creation of washboarding

Washboarding is an undesired effect resulting from the corrugated cardboard manufacturing process and is found on the surface of both face and back liners. If one runs a finger across the surface of corrugated cardboard, a ripple running across the surface, parallel to the flute, can be felt. This ripple occurs when the liner and medium (the flute) are glued together by starch, and the starch shrinks as it dries (Ollett *et al.* [2]). This shrinkage may pull the liner and medium together (figure [1.2.1]). The formation of the geometric shape known as washboarding can be described as a landscape of hills and valleys in succession. The term washboarding is used to describe these undulations because they resemble those of a washboard that was used to clean clothes before washing machines were widely used.

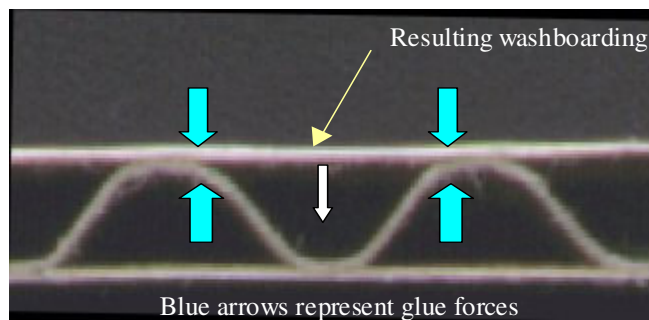


Figure [1.2.1] – Washboard deformation. Image produced by the author.

Single facer board (see figure [1.1.2]) is used in this thesis for various studies using the profilometry techniques described later in Chapter 4. The use of single facer board enables manual bonding of the double backer (back liner) and makes it possible to accurately control and measure the quantity of glue used. This allows experiments that would not be physically or economically viable if a corrugator were to be used. For example, single facer board was used in this thesis to investigate into how the variation of glue quantity affected the properties of the corrugated cardboard.

1.2.2 Washboarding and its relationship to strain

Strain is a vector quantity that is a measurement of the degree of deformation of a material under load or stress. The basic nature of washboarding suggests that at equilibrium in constant environmental conditions, there is a constant stress applied to the liner and fluting by the glue. This stress produces a strain in the liner and the fluting where the strain in the liner is observed as washboarding. The resultant strain may therefore be directly related to the washboard depth. This hypothesis will be tested in this thesis.

Measurement of the washboard profile allows the strain to be calculated. The resultant stress due to the glue can then be estimated if the mechanical properties of the paper are known.

1.3 Measurement of washboarding using digital imaging profilometry

Knowledge of the profile of washboarding is an important requirement for the studies presented within this thesis. Previous methods of measurement or estimating washboarding included a scanning profilometry technique [3], which is precise but slow and laborious, and air leak methods which are quick but imprecise.

A prototype bench top digital imaging profilometer was developed by Reich and Allan [4] and investigated by Gomez [5] for his honours thesis. The method for washboard extraction described by Reich and Allan [*ibid*] resulted in an uncalibrated profile in terms of both x displacement across the sample, and y depth. This method was modified and improved in this thesis to provide absolute measurements by developing a procedure for accurate calibration of the equipment and software. The new instrument designed uses an optical configuration different to the original and is described in Chapter 4.

1.4 Finite element analysis

Finite element analysis (FEA), (also known as finite element modelling) is used extensively in this thesis (see section [3.4]) to analyse washboarding and how the structural performance of corrugated cardboard is affected by the degree of washboarding present.

FEA is a method used for modelling or simulating complex real world material or mechanical systems [6]. It is most useful for non-linear highly complex systems that are impossible or very difficult to solve by analytical analysis. FEA is a discrete method that uses information such as mathematical equations describing real world physical behaviour, look-up tables (e.g. for non-linear material properties), boundary conditions, contact detection of analytical bodies and matrix arithmetic to model real world systems [7, 8, 9].

In the absence of finite element analysis (or other numerical analysis), development of structures must be based on analytical calculations only. For complex structures, the simplifying assumptions required to make any calculations possible can lead to a conservative over design.

Chapter 2 – Literature Review

In order to identify factors that affect the properties of corrugated cardboard and the presence of washboarding, one needs to critically assess each component- from the pulp used to create the paper, through to the properties of the starch used to create the corrugated board.

2.1 Pulping, paper furnish and strength

The strength of a liner depends upon a number of factors but primarily upon

- the pulping process –mechanical, chemical, or a combination of the two,
- fibre types- recycled, short fibre (softwood) or long fibre (hardwood),
- other constituents present, such as fines and lignin which are a function of pulp preparation (refining)

Beating (or refining) alters the mechanical properties of paper. There are a variety of mechanisms involved in changing the properties (discussed below), though their relative importance is the source of some debate [10, 11, 12]. Page [12] suggests that ‘the relative importance of the different beating effects depend on the starting material’.

The mechanical changes during refining involve the compression of the fibres, from a tube, into an oval or flat shape and the removal of the primary fibre wall, increasing the bonding area [11, 13]. The removal of the fibre wall allows water to penetrate the fibre and loosen the fibre structure, resulting in fibre swelling and causes the fibres to soften and become more flexible. This process is referred to as “internal fibrillation” and is often regarded as the most important effect of refining after removal of the primary wall [11]. A further process called external fibrillation involves loosening of the fibrils and production of finer micro-fibrils on the surface of the fibres (also known as “hornification”). This “hornification” causes a large

increase in surface area of the beaten fibres [11, 14] that results in an increase in bonding strength due to the increase in the surface area of the fibres. An increase in the number of fibrils can further strengthen the paper because the fibrils of neighbouring fibres may hook around each other, increasing the fibre-fibre bond strength [11]. Haslach [11] also suggests that beating increases inter-fibre bond formation by allowing more hydrogen bonds to form between cellulose molecules. During beating, fibrillar fines are produced, which can increase bonding between fibres [10, 12, 15] According to research performed by Stratton [16], fines contribute more to bonding than fibrillation in the case of unbleached kraft loblolly pine pulp. Page [12] suggests that fines have a greater effect on the tensile strength of mechanical pulps (accounting for up to 20 % of the increase in tensile strength due to beating), but are of little importance for beaten, dried, low-yield pulps.

As the degree of beating of the pulp increases, the tensile stiffness or strength of the final paper made from the pulp increases, up to the point where cutting occurs as the fibres begin to break [13, 14]. This can result in no further increase, or in some cases, a reduction in the strength of the paper [17]. A study by Dasgupta [18] confirms these observations. His work using a pulp of ratio 70/30 northern softwood kraft and chemithermo-mechanical pulp found that the increase in tensile strength due to beating was a result of an increase in the number and frequency of fibre-fibre bonds. If pulp beating was continued for an extended period of time, a decrease in tensile-strength was observed, which was largely due to damage to fibres. Wistara and Young [15] also found the same result for never-dried, mechanically treated blackwood spruce kraft pulp, where an increase in beating resulted in a decrease in the tensile strength.

Common fibre types that are used in commercial manufacture include virgin short fibre (hardwood, such as eucalypts), virgin long fibre (softwood ie. conifers) and recycled fibres,

which contain different quantities of long and short fibres as well as damaged fibres.

Recycled fibres have been found to not perform as well as virgin fibres in conditions of cyclic humidity [19, 20, 21]. Wistara and Young [15] found that the tensile strength of untreated kraft pulp decreased by 37 % when recycled, and reduced to 50 % when recycled for the second time. A significant reduction in the strength and survival time has also been observed for boxes manufactured with recycled pulp [22]. The type of wood from which the pulp is derived (eg. softwood or hardwood) has also been shown to have an affect on the properties of pulp when recycled [15]. For example the tear factor of bleached kraft hardwood pulp has been found to decrease when recycled, whereas it has been observed to increase for recycled bleached softwood kraft pulp [15]. High-yield pulps seem to be less affected by recycling, and hornification is believed to be more prevalent in the low-yield pulps [15]. This may be due to the higher levels of lignin present in high-yield pulps [11] which is believed to be a factor involved in the changes in the properties of recycled pulp [15].

2.2 Thermal, Moisture and Mechanical Properties of Paper.

2.2.1 The effect of relative humidity and temperature upon the moisture content of paper

It is well known in the paper industry that if the temperature of a piece of paper remains steady, the moisture content in the paper increases with increasing relative humidity (RH) [11, 23, 24]. Figure [2.2.1] from Benson [23] shows a typical relationship between the relative humidity of the environment and the moisture content of a softwood kraft-liner board at 23 °C. The moisture content of paper increases from 4 % to 9 % (grams of water per grams of dried fibre) as the relative humidities increase from 20 % to 70 %. The rate of moisture uptake for a rise in relative humidity in paper is significantly higher above 70 % than it is below.

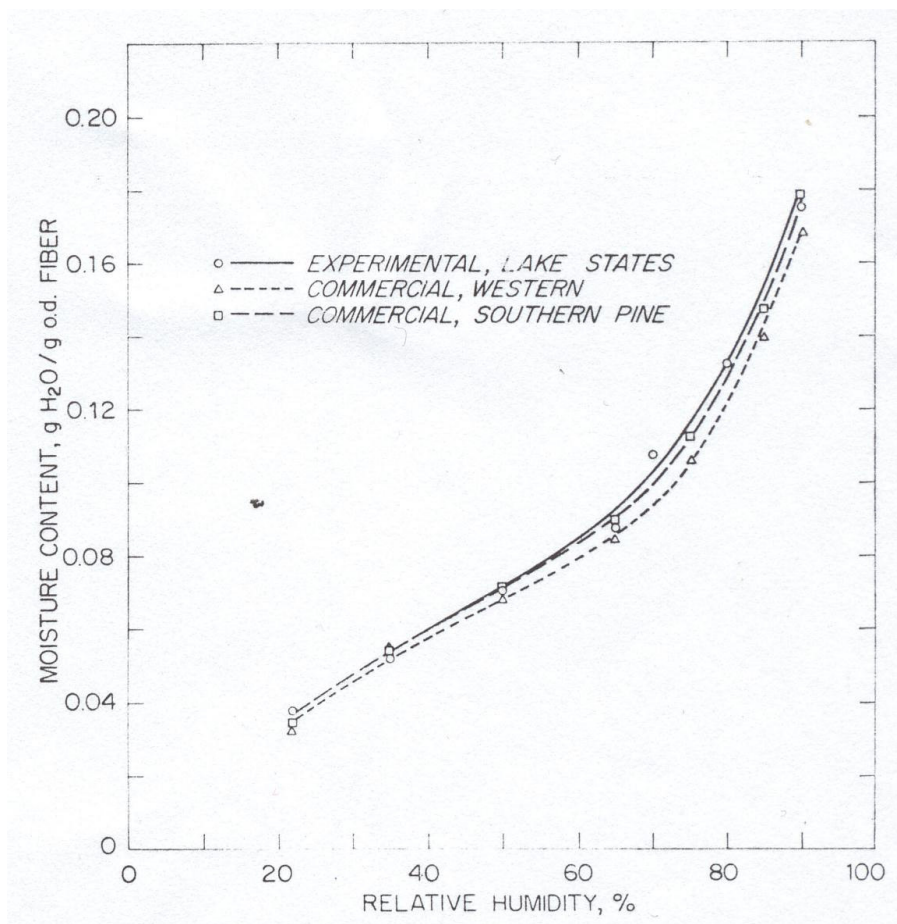


Figure [2.2.1] – How moisture content of kraft-liner board varies with relative humidity, reproduced from Benson [23].

The trend for kraft-liner board illustrated in Fig [2.2.1] is also observed for sorption isotherms of cellulose by Wink [24]. Sorption isotherms represent the relationship between RH and

moisture content of a material at diffusional equilibrium and constant temperature. The sorption isotherm for cellulose presented in Wink's paper [*ibid*] suggests that the moisture content of cellulose also increases more rapidly above 70 % RH.

Benson [*ibid*] also studied the change in moisture content as temperature increases and he found that there is a linear relationship between temperature and moisture content of the paper (softwood kraft-liner board) at relative humidities of 50 and 65 %, illustrated in figure [2.2.2] below. It was also shown that as the temperature increased, moisture content decreased. Skogman and Scheie [25] observed a similar relationship between moisture content and temperature for kraft paper between $-20\text{ }^{\circ}\text{C}$ and $20\text{ }^{\circ}\text{C}$ at a relative humidity of 30 %. However, in his review Haslach [11] notes that the results presented by Skogman and Scheie [*ibid*] contradict Benson's findings at higher relative humidities, as the moisture content was observed to increase with increasing temperatures from $-20\text{ }^{\circ}\text{C}$ up to approximately $5\text{ }^{\circ}\text{C}$.

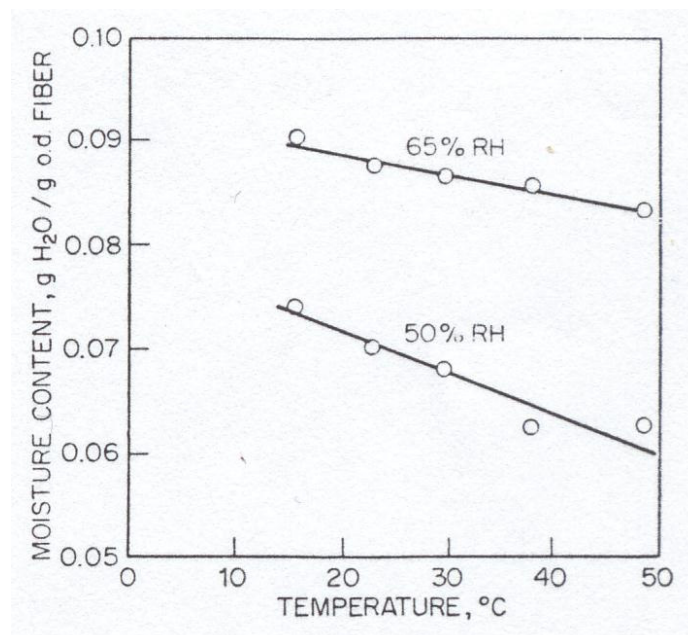


Figure [2.2.2] – Moisture content of softwood kraft-liner board in constant humidity in an environment of changing temperature, reproduced from Benson [*ibid*].

With an increase in moisture content there are significant changes in the fibre-fibre bonding within the paper, which alter the paper's properties [26, 27]. Zauscher *et al.* [27] proposed that until the paper reaches a critical water level, as the moisture content increases, each water

molecule can break one hydrogen-bond. Once the moisture content of the paper exceeds this level, water can break several hydrogen-bonds simultaneously, affecting the elastic modulus (see section [2.2.6]).

The rate of moisture uptake or loss is different depending on whether relative humidity is rising or falling. Hence the moisture content of paper will depend on the previous state of the paper moisture content and if the humidity is rising or falling ie. the paper experiences moisture hysteresis [28, 29].

For these reasons, paper testing is usually performed at ISO conditions (50 % RH and 23 °C) [30] after pre-conditioning at low humidity.

2.2.2 Moisture and hygroexpansion

As the moisture content of paper increases, fibres swell and paper expands in each of the MD, CD and z directions [31] where the expansion in the CD and z-directions are similar [31, 32]. De Ruvo *et al.* [31], Enomae and Lepoutre [33] and Forseth and Helle [34] have all observed the expansion of paper fibres in the presence of water. The expansion of paper is anisotropic with expansion in the machine direction (MD) significantly less than that in the cross direction (CD), [32]. Expansion in the z-direction involves an increase in thickness of the liner

Chalmers [35] studied the effect of humidity upon the elastic modulus for unbleached softwood kraft liner and recycled fibre medium and found that at relative humidities higher than 50 %, the elastic modulus of paper in compression is significantly higher than in tension, and that as the relative humidity increased, the elastic modulus decreased. Benson [23] suggested that the reduction in tensile properties observed at relative humidities of 50 % or

higher may be a result of relaxation of the stresses present in the paper as a consequence of manufacturing. As the paper dries, stresses (often in the form of wrinkling and micro-compressions) can occur between fibre bonds that had begun forming while the fibres had been immersed in water [11]. This can largely be reversed by increasing the moisture content of the fibre bond (ie. by increasing the relative humidity) [11].

Benson [*ibid*] has also shown that changes in moisture content of 1 % causes an increase in thickness of 1.5 % for the kraft-liner board he investigated. Soremark and Fellers [36] found that the thickness of corrugated kraft-liner board increased by 0.52 % when the humidity was increased from 40 to 80 % RH. The lower percentage increase recorded by Soremark and Fellers may potentially be attributed to damage to the fibres which limited swelling (and therefore thickening), as a result of the corrugating process [36].

Enomae and Lepoutre [*ibid*] have shown that the expansion rate in the presence of water varies depending on the paper furnish. While Enomae and Lepoutre's study [*ibid*] was only performed with a small number of samples, the results indicated that the ability of fibres to return to a tubular shape when soaked in water differed between samples obtained from never-dried pulp, defibrated handsheets and beaten dry pulp.

2.2.3 The Measurement of Paper Thickness

The z-direction expansion may be measured directly using a micrometer for given moisture levels, but the measurement (as performed in this thesis) may become erroneous due to two effects. One is that in the act of mechanical measurement of the thickness (the norm), a pressure is applied. This pressure compresses the paper, therefore reducing the measured thickness. How much this affects the measurement may depend on the elastic modulus of the paper and the pressure applied. This is a significant problem when thickness is measured at

different relative humidities due to the changes in the mechanical properties of the paper. An increase in moisture content tends to expand the paper (de Ruvo *et al.* [31]) and this may also contribute to the difficulty in obtaining a reliable measurement of the thickness of a paper sample.

Another potential source of error is due to the uneven surface of the paper because of protruding fibres, fibre flocks and fibre gaps [37]. As a result, the mechanical measurement may over-estimate the mean thickness, though this over-estimate should stay consistent irrespective of changes in the ambient relative humidity.

The difficulties of measuring paper thickness can also affect estimates of other properties such as bending stiffness.

The bending stiffness (B) of a material is proportional to the cube of thickness (T) [38], ie.

$$B \propto T^3 \quad \text{Equation [2.2.1].}$$

The increase in bending stiffness may be underestimated for a given increase in relative humidity due to the underestimation of the thickness as measured mechanically.

An increase in moisture content can increase the thickness of the paper with a corresponding non-linear increase in bending stiffness as given by the above relationship. However, opposing this increase may be a linear decrease in elastic modulus as the moisture content increases [39].

2.2.4 Mechanosorptive (accelerated) creep of paper

Paper creeps because it behaves like a semi-amorphous liquid and can therefore flow below its glass transition temperature [17, 40]. There is no standard method for measuring creep,

though a variety of techniques have been summarised by Panek *et al.* [41].

When a semi-amorphous material such as paper is under a constant stress, the material stretches relatively quickly (this is also called the initial strain). After the initial strain, the material continues to stretch or creep at a much lower rate [36, 42]. Under constant stress, and in a constant environment, the rate of creep is lower than when paper is subjected to dynamic environmental conditions [43]. For example, if the paper under investigation is in an environment where humidity is cycled, the creep rate may accelerate due to repeated stretching and relaxing of the paper for a given load [44].

The stretching and relaxation of the paper occurs because the stiffness of paper changes with humidity. Because the paper stress and strain relationship is non-linear, the paper may retain a permanent plastic strain as this cycling of humidity continues. It is also believed that stresses between fibre bonds due to moisture cycling causes mechanosorptive creep [11, 43].

Fibres swell differently in the transverse (usually CD) and longitudinal (usually MD) directions [11, 43], (see section [2.2.5] for details), causing stresses between fibre bonds when the moisture level is altered [43]. Alfthan *et al.* [43] and Habeger and Coffin [45] suggested that these differences in swelling may also be the result of moisture gradients and/or heterogeneities in the paper. During sorption, moisture gradients are formed through the paper, which in turn forms swelling gradients, and therefore stress gradients. Different paper layers may swell differently when the moisture levels change, therefore producing additional stresses.

The creep of a material is typically logarithmic with the bulk of the creep occurring at the start of stretching [11, 45, 46]. The strain at which a paper may break due to creep can be as much as four times that seen for the strain to failure in an elongation test for determining a paper's

stress versus strain curve [11, 46]. The exact point at which the paper will break is difficult to predict.

The paper furnish as discussed in section [2.2.2] has an effect on how a paper creeps. Zhang *et al.* [47] were able to demonstrate that lignin-containing high-yield pulp produces the smallest amount of creep strain in comparison to pulps containing high levels of either hemicellulose or cellulose. Several studies have shown that recycled fibres do not perform as well as virgin fibres in a cyclic humidity environment [19, 20, 21]. This can become a major problem for boxes that are to be used in such environments and for this reason, it is preferable for virgin fibre to be used in particularly harsh environments of high humidity or large cyclic variations in humidity [19].

2.2.5 Anisotropy of paper

In the process of paper making, fibres have a tendency to align in the machine direction (MD) of the paper [32]. However, Loewen and Foulger [48] indicate that fibre alignment is not always exactly in the MD due to cross-flows as paper moves from the wet-end to further down the paper making machine. The degree of misalignment of the fibres from the MD is known as the tensile stiffness orientation (TSO) angle [48]. As the fibres dry, they shrink anisotropically by up to 30 % in the transverse direction (CD) yet only 1-2 % longitudinally (MD) [49, 50]. The mechanical properties of the liner therefore usually differ with orientation [23, 51]. It has been shown that, when TSO approaches 0 ° the stiffness of the paper progressively decreases as the angle with respect to MD is increased [48, 51]. In this situation, ninety degrees to MD (ie. the cross direction (CD)) is usually the most compliant orientation [48, 51].

The tensile [23] and compression [35] properties of paper are also orientation dependent. These properties are in addition affected by changes in the moisture content, due to changes in the relative humidity and/or temperature of the environment [23]. Benson [23] (see figure [2.2.3]) showed how the tensile properties (specifically the stress-strain relationship) changed with changes in relative humidity from 22 % to 90 % RH. The paper was a softwood kraft-liner board and was tested in both machine direction (MD) and cross direction (CD) at a temperature of 23 °C.

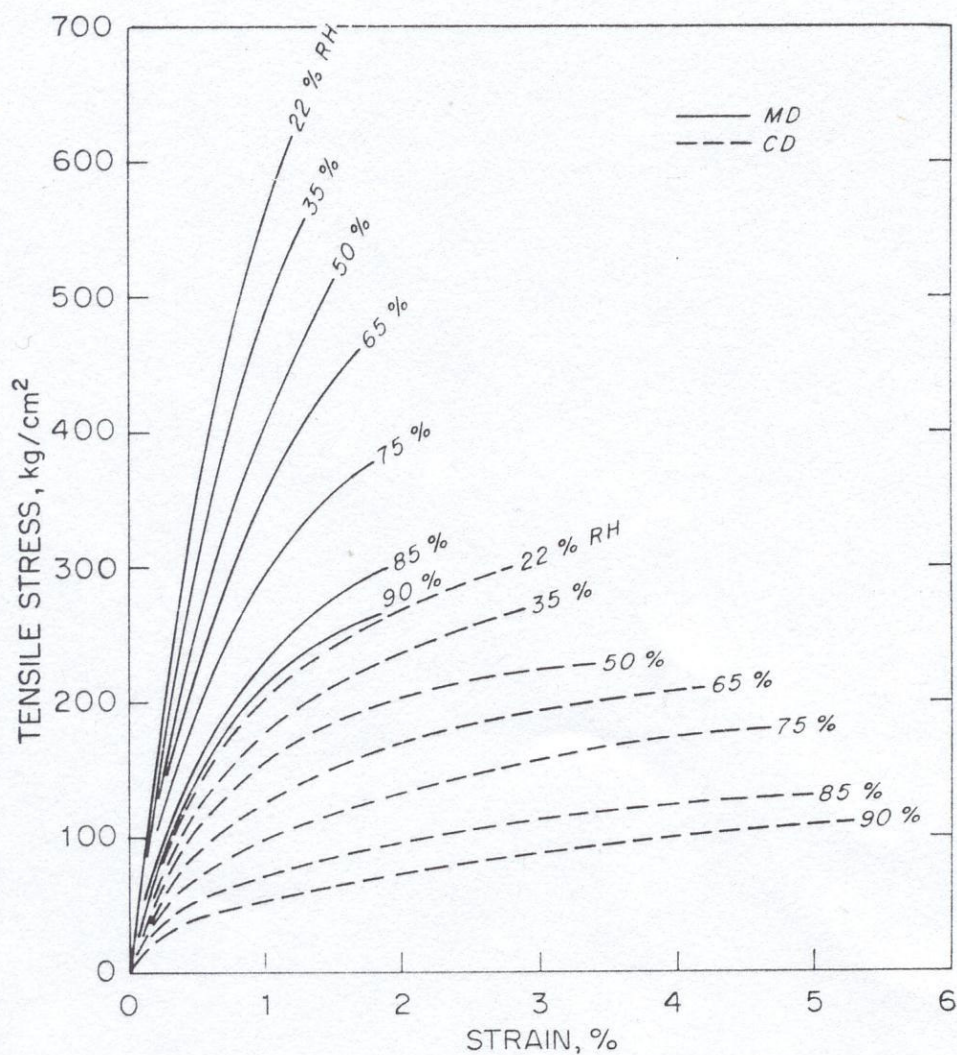


Figure [2.2.3] – Anisotropic stress-strain relationships of paper in different relative humidities, reproduced from Benson [ibid].

It can be seen from figure [2.2.3] that the stiffness of paper in CD is lower than that in MD for the relative humidities shown, and as the relative humidity of the environment increases (and therefore the moisture content of the paper increases) the stiffness of paper decreases.

An alternate explanation for the anisotropy of paper under a load is discussed by Haslach [11]. The work of Edge (presented in Haslach [*ibid*]) suggests that the anisotropy of paper is largely a result of tensile stress in the MD which draws the paper through the machine. This stress increases the MD strength of the sheet and reduces compliancy. Edge's work also suggests CD tensile strength is greater in the centre of the paper web due to "stresses induced by the friction from transverse shrinkage of the sheet against the drier drum". However, this may not account for how TSO differs across the web whereas the fibre alignment theory presented by Loewen and Foulger [*ibid*] does.

2.2.6 Sonic modulus, anisotropy of paper and relationship to elastic modulus

The elastic modulus of a material is defined as stress (force per unit cross-sectional area) divided by strain (change in the shape of the material) [52]. There are two general methods used to determine the elastic modulus of paper; tensile measurements (load-elongation, also known as Young's modulus) or ultrasonic measurement (sonic modulus). Load-elongation measurements involve a sample being placed between the jaws of a tester, and application of a constant strain-rate. The degree of deformation is then measured [27, 53]. Ultrasonic measurements involve measuring the speed of sound (stress and strain propagation) through a material [27].

Figure [2.2.4] displays a polar plot of the square of speed of sound as a function of angle for a typical paper as measured by the author. The elastic modulus, as measured ultrasonically is directly proportional to the square of the speed of sound through the material [54] and

therefore the polar plot is an indication of how the sonic modulus changes with fibre orientation due to the anisotropic behaviour of machine manufactured paper [11, 48, 51]. Walter *et al.* [55] found the velocity was on average 1.48 times faster in the MD than in the CD at 1 MHz.

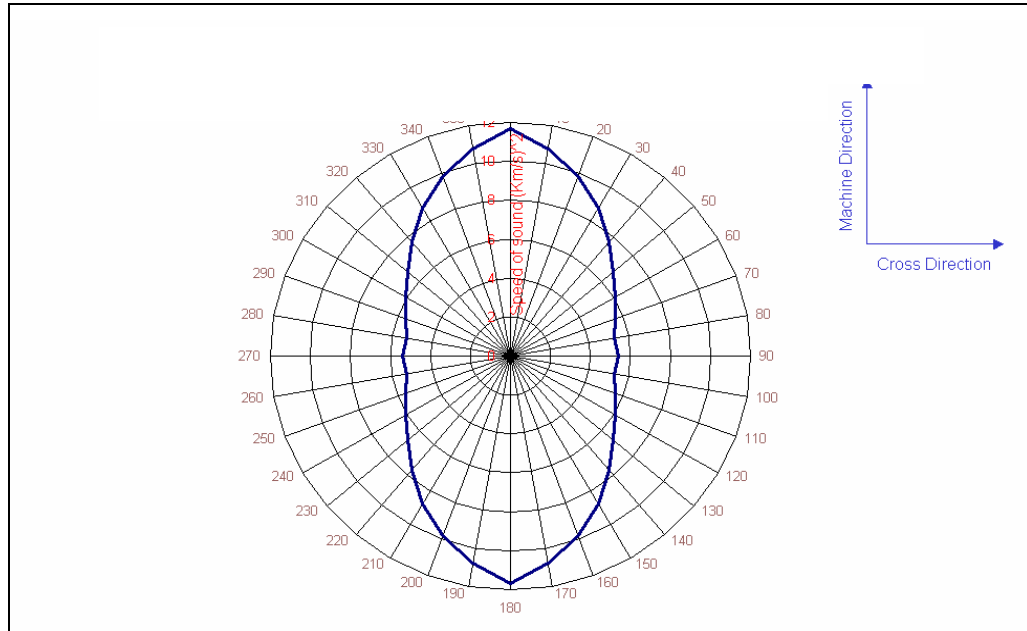


Figure [2.2.4] – Polar plot of the speed of sound squared for a typical machine manufactured paper measured by the author using a Nomura Shoji sonic sheet tester at 50 % RH and 23 °C.

An advantage of determining the elastic modulus via ultrasonic measurement (as compared to using load-elongation methods) is that it minimises disturbance of the paper and can be used to assess the tensile properties of paper non-destructively. Ultrasonic measurement of the elastic modulus has successfully been used for on-line measurements by Walter *et al.* [55] and Ridgway *et al.* [56].

On the other hand, an advantage of load-elongation methods is that it provides average of the whole area between the machine jaws, whereas ultrasonic methods only provide the modulus of the fastest route to the detector. However, elastic modulus as measured by ultrasonic methods has several advantages over load-elongation testing including:

- Less room for experimental error, as the machine jaws may slip during testing of Young's modulus [53]
- Minimisation of the relaxation phenomena (discussed below), which is more evident in mechanical measurements [27, 53] due to longer stress durations
- The influence of the relaxation phenomena is significantly higher at high RH for mechanical testing [39]

Since the tensile stiffness measurements vary according to the strain rate, one can argue that ultrasonic measurements are just one point “on a continuum of measured stiffness properties” [53].

Ultrasonic measurement differs to the elastic modulus measured by load-elongation generally by a minimum increase of approximately 10 %, where the average increase is typically ~ 20 %, and the reason for this difference is not apparent [57]. According to Zauscher *et al.* [39], at zero water content, the difference between the ultrasonic and Young's modulus is approximately 10 %. As the moisture content increases, the difference between the elastic modulus measured ultrasonically and Young's modulus also increases [53]. Coffin *et al.* [53] have suggested that this is due to plasticisation of the amorphous components of paper (also known as the relaxation effect), which is more obvious in longer duration tests such as the load-elongation test. According to Coffin *et al.* [*ibid*] the ultrasonic measurements may be different to the mechanical (load-elongation) measurements by the Poisson factor (due to different stress rates) which also works out to be about 10 %.

Zauscher *et al.* [27, 39] suggested that the difference between the elastic modulus measured via ultrasonic and load-elongation methods may be due to different types and densities of hydrogen bonds being activated. They propose that ultrasonic methods only measure the

modulus of the stiffest hydrogen bonds (as the waves travel quickest through them) whereas the mechanical method measures the average of the bonds held between the machine's jaws.

However, as Coffin *et al.* [53] point out, paper is a complex material, and is unlikely to be governed solely by hydrogen bonds. The data presented by Coffin *et al.* [*ibid*] suggests that the difference between the elastic modulus measured using either sonic or Young's modulus may be a result of the difference in time taken to undergo the two methods. Paper is known to exhibit a time-dependent response [11, 53] and as load-elongation tests typically take ~ 0.1 seconds, whereas ultrasonic tests take in the order of 20 microseconds [53] it seems logical to assume these two tests will yield different results

.

2.3 The Effect of Moisture upon the Properties of Starch

Starch is the most commonly used adhesive for corrugated board [58, 59]. The properties of starch are known to have a significant effect on the performance of corrugated boxes [59]. These properties are often overlooked in studies of corrugated board, especially in the case of washboarding, where there has been little published information.

Starch is a polymer of α -D-glucose composed of amylose, a straight-chain polymer of α -glucose and amylopectin, which is a highly branched polymer of α -glucose [60, 61]. For the paper industry, starch is typically derived either from wheat, potato or maize (corn) [58, 60]. There are detectable differences in the properties of these starches and an in-depth review has been presented by Ellis *et al.* [60].

It has been shown that the moisture content influences the properties of starch [62]. Figure [2.3.1], reproduced from Kirby *et al.* [*ibid*] shows how the elastic modulus of wheat starch varies with moisture content.

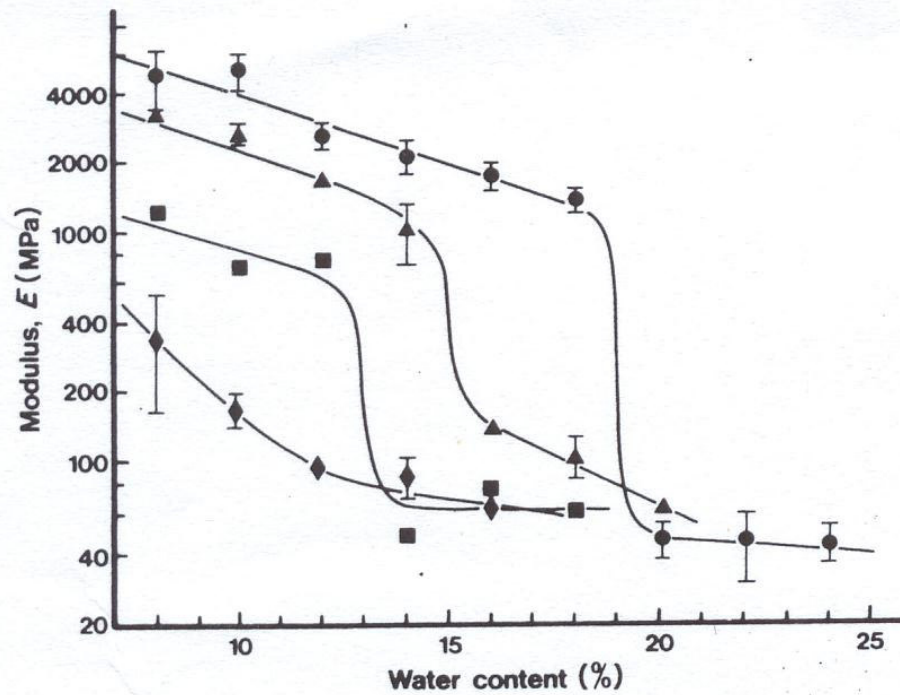


Figure [2.3.1] – Change in elastic modulus of various wheat starch - polymer combinations for different moisture contents, reproduced from Kirby [62]. Composition: (●) wheat starch; (◆) wheat starch-xylitol; (■) wheat starch-glycerol

Kirby *et al.* [ibid] found that the elastic modulus of pure wheat starch drops linearly from approximately 5.0 GPa at 7 % water content to 1.8 GPa at 18 % water content, after which a sharp drop is seen from 18 % to 20 % water content. This sharp drop is thought to be due to a glass transition (Ollett *et al.* [2] and Shen *et al.* [63]). Willet and Doane [64] also found that the tensile strength and modulus of a corn starch composite decreased as the moisture content increased above 6 %, while the strain to break increased. Willet and Doane [ibid] suggested that the changes in the mechanical properties of corn starch observed above 7 % is partly due to the observed decrease in the crystallinity of the starch granules, and have demonstrated that water is an effective plasticiser of starch.

Ollett *et al.* [ibid], Kirby *et al.* [ibid] and Willet and Doane [ibid] have shown that the moisture content of the starch varies with the relative humidity of the surrounding environment. This is important in describing the behaviour of washboarding in a cyclic

humidity environment. Ollett *et al.* [ibid] and Kirby *et al.* [ibid] measured moisture content directly by a gravimetric method and did not publish the relative humidities of the environment, but stated which saturated salt baths were used to control humidity. The relative humidity was then estimated as part of the work for this thesis from known previously measured saturated salt bath humidities [65]. The relationship between relative humidity and elastic modulus can be used to approximate the change in glue force between the liner and fluting for increasing humidity, and is described in section [3.2].

2.4 Washboarding

The structural integrity of a box made of corrugated cardboard is believed to be compromised by washboarding of the liners [66] while washboarding has also been shown to create label adhesion problems [67]. It is also thought that washboarding inhibits the uniform application of ink when printing on corrugated cardboard unless sufficient pressure is applied [68]. In this case the pressure required may mean that the box structural integrity can also be diminished by the destruction of the internal fluting [66].

There is a general lack of published methods on how to quantitatively assess washboarding and its effects, as was noted by McGrattan [69]. Zang and Aspler [70] also commented “there is no published method on how the degree of washboarding can be quantitatively evaluated”

The basic nature of washboarding suggests that for constant environmental conditions, there is a constant stress applied to the liner and fluting by the glue [66]. This stress produces a strain in the liner and the fluting where the strain in the liner is observed as washboarding. The resultant strain may therefore be directly related to the washboard depth.

Snyder [59] proposed that washboarding is often the result of improper handling of starch adhesive. When the starch adhesive is not applied consistently, the moisture content of the corrugated board is also inconsistent, and textural changes on the surface begin to occur. The setting process is achieved by evaporating water from the adhesive, however once the double backer has been applied and the flutes have been formed, the use of significant pressure to force heat through the board would crush the boards. To ensure bonding, the sheet is pulled through a hot plate section, which takes much longer to dry, allowing more stress to build up.

Laschitz [71] discussed the effect of condensed moisture on the inside of a liner. There is a certain amount of condensation on the liner due to the heating of the glue, which releases moisture from the glue as steam, and is deposited on the inside of the liner. He claimed that washboarding is due to the expansion of the liner as a result of this moisture condensation on the paper which releases the stresses introduced when the liner was manufactured.

When the moisture of the liner returns to equilibrium, depending on the relative humidity, compression forces begin to develop (as a result of drying of the liner and fluting), causing washboarding. Laschitz [*ibid*] neglected to mention anything about the effect of the shrinking of the glue, and was possibly unaware of this being a major cause of washboarding, as he stated that he did not know why washboarding is always to the inside, and never to the outside.

Lashitz [*ibid*] also found that paper with different fibre orientation produced different degrees of washboarding. However, this conclusion was drawn from experiments performed with only two papers tested with differing fibre alignment..

Laschitz [*ibid*] made some recommendations to paper manufacturers, suggesting that short compression testing should be performed at lengths equal to the period of the fluting to keep this value as high as possible, to produce a low washboarding liner. However, this would be analogous to measuring the bending stiffness of the liner as opposed to the compression strength, but these values should still represent how much washboarding would be produced for certain manufacturing conditions.

Collier [72] discussed the use of starches with a high solid percentage to improve the quality of corrugated boxes. He found that the use of high solid Minocar glue eliminated

washboarding for the F-flute (a microflute) used in his study. However, due to the small pitch between the fluting tips observed for F-flute (see table [1.1.1]), the washboarding was expected to be negligible. There was no analysis of washboarding, other than stating that washboarding was completely removed if the solid content of the glue is 37 % (the content of Minocar glue) compared to a normal range of 22 to 28 % solid content. However, the use of high solid content (ie. low moisture content) as Collier [72] stated, may result in insufficient glue transport, incomplete gelling, and de-watering too quickly may lead to glue setting prior to penetration of the paper (creating brittle bonds).

It is difficult to conclude from this paper the relative effect the increase in solid content would have on washboarding for fluting of greater pitch (such as B or C-flute), however the increase in solid content would probably decrease glue shrinkage, and hence this supports the view that the shrinking glue creates the washboarding. It would be worthwhile repeating this study for larger flute types.

Netz [3] used a scanning laser to measure the surface profile of a corrugated board. The laser sensor travelled across a board at a speed of 100 mm/s with a sampling frequency of 125 Hz, with measurements made using a sampling distance of 0.8 mm. Such coarse sampling would typically give only 7 to 8 points from tip to tip of a flute and may underestimate the absolute washboarding depth as the lowest point may easily be missed.

Knowledge of the shape of the washboarding may be used for modelling purposes. An understanding of how geometry affects the structural properties can be used to ensure that a model provides correct results. One aspect not yet published is the shape of the valleys, which is investigated in this thesis.

2.4.1 Washboarding and the printing quality of corrugated cardboard

The ability to print onto a corrugated cardboard substrate is often influenced by the extent of washboarding [14, 68, 70]. Printing a company logo on the side of a box is important to corrugated box manufacturing customers who use these boxes to ship and advertise their goods [66, 70]. There is an increasing trend towards the use of advanced graphics in the packaging and box industries. Coleman [73] commented that the use of graphics and print on boxes is expected to grow in the next 5 years (2001-2006) as shipping containers become “display and point of sale” containers. Kamp [74] also points out that corrugated boxes account for ~ 30 % of the flexographic printing market worldwide. Coleman [*ibid*] stated that “minimisation of the washboarding phenomenon is an added requirement to be met by a flexographic thin-plate system.”

If a less than adequate pressure is used in printing the logo for a given washboard depth, stripes may appear in the printed picture [75]. In full tone printing, this is due to ink not reaching the bottom of the washboarding valleys, resulting in stripes of ink on the hills of the washboarding and the absence of ink in the valleys [68]. Boards unaffected by striping receive ink on both the hills and in the valleys. Images of stripes on the surface of corrugated board are shown in Figures [2.4.1 – 2.4.3] and are from samples used in this thesis (see section [3.5]). The figures show the full range of quality of full tone print on corrugated board from severe to negligible striping.

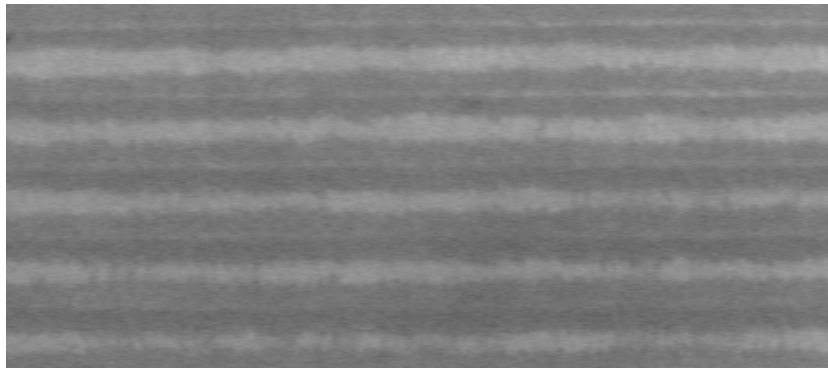


Figure [2.4.1] – Full tone printing – severe striping (printed at ISO conditions by the author)

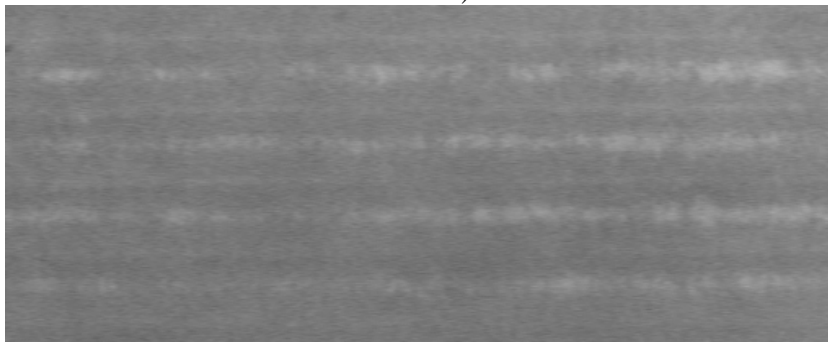


Figure [2.4.2] – Full tone printing – partial striping (printed at ISO conditions by the author).

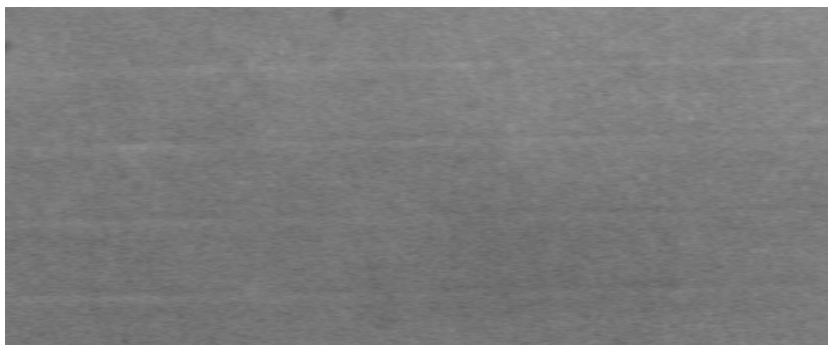
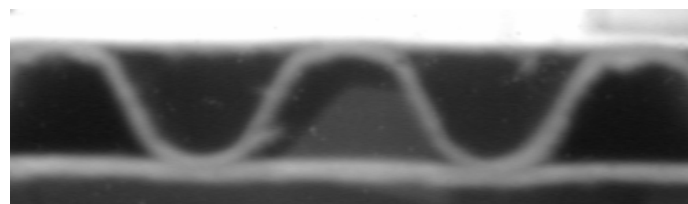


Figure [2.4.3] – Full tone printing – negligible striping (printed at ISO conditions by the author).

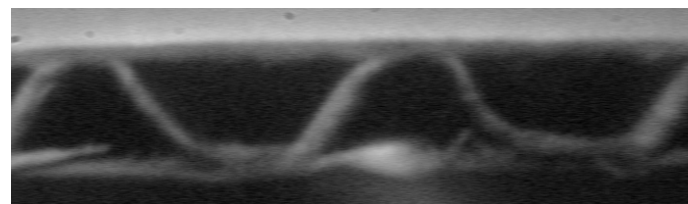
The severity of the striping effect is related to the degree of washboarding present [73, 75, 76]. It is possible to make ink reach this region by increasing the pressure, however this can cause the corrugated cardboard to be crushed (see Fig [2.4.4]) and lose a great deal of its structural strength.

Netz [3] however concluded that the effect of washboarding upon printing is minimal because

corrugated cardboard may be compressed by up to 400 μm . He stated that print nip, fluting, stiffness, and thickness differences have a greater influence on the print quality. He commented that washboarding may influence print quality if the corrugated board is insufficiently compressed and recommended against reducing print pressure to avoid striping due to washboarding, deeming that it is possible to have full board coverage if enough pressure is applied while printing, even though the washboarding depth is large. However, if there is an excessive amount of washboarding, the fluting is more likely to be crushed (as illustrated in Fig [2.4.4]) due to this increased printing pressure.



(a) Not crushed.



(b) Crushed

Figure [2.4.4] – Crushing of corrugated cardboard fluting. Images produced by the author.

A study by Selway and Kirkpatrick [77] showed that the compression of fluting material significantly reduced final box strength, and its survival time under load, and therefore it is advantageous to keep printing pressure to a minimum. The minimum can then be determined by the amount of washboarding present in conjunction with the required box strength and/or survival time required.

Another commonly used method of printing onto liner substrate is printing on a corrugated cardboard box before manufacture (known as preprint) [70]. This largely eliminates the effects of washboarding on print quality. However, washboarding may change the reflection

characteristics of the final product and hence the appearance due to the geometric surface variation.

Stolpe [66] writes “the development of the preprint was foreseen some years ago and has not fulfilled expectations in how often it used. This is due to long production runs required to motivate an involvement of a 2.5 metre wide flexographic printing machine in conjunction with the increasing demand for short delivery times, short production lots and just in time deliveries.”, indicating that it is preferable to use post production printing.

2.4.2 Quantifying print quality

Carabin and Kerr [78] and Lyne [79] studied in their respective papers, methods of determining the relative effects of a variety of factors on print quality. Carabin and Kerr set out to obtain an equation to describe print quality preferences by having a variety of newsprint samples ranked by a large number of people. Their study found that printed and unprinted sheet colour preference correlated highly, and other paper qualities such as roughness failed to improve this relationship. Lyne suggested the use of a statistical technique to establish the relative importance of factors involved in assessing print quality. Multi-dimensional scaling allows for the separation of the main factors used by individuals when judging print quality, and can also be used to correlate these subjective print quality variables to corresponding physical properties of both prints and paper.

Comparatively, Netz [3] primarily looked at the effect of washboarding on print quality. Netz [*ibid*] used a subjective method to analyse the degree of washboarding, as he stated that visual inspection was the most reliable in comparison to imaging methods he had at his disposal, which would only characterise small areas (area smaller than 5 mm²). He stated that it was important to evaluate large areas for a more “macro” analysis. In this thesis, a method was

developed to analyse the degree of striping by the use of image analysis of a sample area of approximately 40 cm² (see section [3.5]).

Netz also showed that an increase in liner grammage reduces the appearance of striping. He stated that this was due to a higher liner grammage having a smaller pressure difference between the liner along the flute tips and the areas in between. However, he did not relate this to the decrease in washboarding as liner grammage increased, see figure [2.4.5].

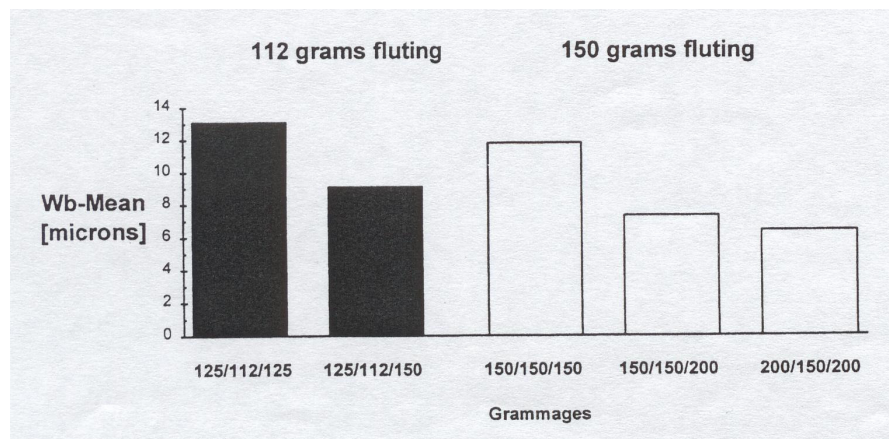


Figure [2.4.5] – From Netz [3] showing a decrease on washboarding for increasing grammage

It should be noted that washboard depth, for the 200 g/m² white liner, would be significantly less than for the 125 g/m² white liner as the bending and tensional stiffness would be expected to have increased.

2.4.3 Finite element analysis (FEA)

FEA is a computer-based numerical technique for calculating the strength and behaviour of engineering structures. It can be used to calculate deflection, stress, vibration, buckling behaviour and many other phenomena [7, 9]. It can be used to analyse either small or large-scale deflection under loading or applied displacement [8].

FEA has been used to study bending stiffness of corrugated board [80]. FEA analysis has been used extensively in the thesis of Nordstrand [6] to model the structure and behaviour of corrugated board and boxes.

Chapter 3 – Theory

3.1 An overview of Fourier analysis.

Fourier methods were used in this thesis for extracting the washboarding profile from a digital image to determine the shape of washboarding. Fourier analysis is a well known and established method of which a short overview is presented next. There are many good texts [81, 82, 83] available if a more thorough understanding is sought.

The Fourier transform defines a relationship between a signal in the time domain and its representation in the frequency domain. Being a lossless transform, no information is created or lost in the process, so the original signal can be recovered from knowledge of the Fourier transform, and vice versa.

3.1.1 Fourier series expansion

The Fourier transform, in essence, decomposes or separates a waveform or function into sinusoids of different frequencies, which, when summed, recover the original waveform

The Fourier series for a periodic signal with length T and period ω and time t is given by,

$$f(t) = a_0 + \sum_{n=1}^{n=\infty} a_n \cdot \cos(n \cdot \omega \cdot t) + \sum_{n=1}^{n=\infty} b_n \cdot \sin(n \cdot \omega \cdot t) \quad \text{Equation [3.1.1]}$$

Where,

$$a_n = \frac{2}{T} \int_{-T/2}^{+T/2} f(t) \cdot \cos(n \cdot \omega \cdot t) \cdot dt \quad [\text{for } n > 0] \quad \text{Equation [3.1.1a]}$$

$$b_n = \frac{2}{T} \int_{-T/2}^{+T/2} f(t) \cdot \sin(n \cdot \omega \cdot t) \cdot dt \quad \text{Equation [3.1.1b]}$$

and a_0 is a special case and is equal to

$$a_0 = \frac{1}{T} \int_{-T/2}^{+T/2} f(t) \cdot dt \quad \text{Equation [3.1.1c]}$$

The amplitudes, a_n and b_n , in equation [3.1.1] may be calculated by using the integral Fourier transforms as seen in equations [3.1.1a-c]

These equations describe how a well-behaved function (monotonic and continuous) can be decomposed into a summation of sines and cosine functions of various component frequencies.

3.1.2 Real and imaginary terms

The original shape or function can be reconstructed by the addition of any existing sine (real) and/or cosine (imaginary) numbers as is shown in equation [3.1.1]. Various approximations to the original function or shape can be defined if certain frequencies are omitted from the addition ie. if high frequency components are rejected, then the derived curve or shape will be a low pass approximation to the original function, and if the low frequency components are not included then the reconstructed function becomes a high pass approximation

3.1.3 Multiplication rule

The transformation of a function multiplied by a constant (A) is equal to the transformation multiplied by the same constant, ie.

$$FT(A \cdot f(x)) = A \cdot FT(f(x)) \quad \text{Rule [3.1.1]},$$

where $FT(f(x))$, is the Fourier transform of the function $f(x)$. This means that varying the amplitude of the original function, while retaining the overall shape, will not change the ratios of the frequency components.

3.1.4 Phase determination of a specific frequency

The phase of the original function may be calculated by using,

$$\theta = \tan^{-1}\left(\frac{a_n}{b_n}\right) \quad \text{Equation [3.1.2].}$$

This equation is primarily used in this thesis for extracting a washboarding profile from a digital image (see section [4.1])

3.1.5 Discrete Fourier Transform (DFT)

The equations [3.1.1a-c] relate to the transformation of a continuous waveform, however, the waveforms found in digital analysis, as investigated in this thesis, are not continuous but are discrete. These equations are altered slightly by the replacement of the integral sums with discrete summations. This Fourier transformation method is termed as the discrete Fourier transform (DFT) and the theory described for the continuous function remains valid for a DFT.

3.1.6 Fast Fourier Transform (FFT)

The discrete Fourier transform (DFT) is slow for large samples as the time taken for one transform is proportional to the sample size squared, ie.

$$T_{FFT} \propto N^2 \quad \text{Equation [3.1.3],}$$

where T_{FFT} is the time taken per transform and N is the sample size transformed.

A method that reduces the time taken is the Discrete Fast Fourier Transform (DFFT) most often referred to as the Fast Fourier Transform (FFT).

The time taken for a given sample size, for a FFT, is proportional to

$$T_{FFT} \propto N \cdot \log(N) \quad \text{Equation [3.1.4],}$$

where T is the time taken per transform and N is the number of samples in a waveform to be

transformed. The FFT was developed as a fast algorithm that greatly reduces the required time for larger sample lengths. The FFT is based on optimising the DFT by a reduction in calculations that are repeated in a DFT.

3.2 The effect of relative humidity upon wheat starch (glue)

3.2.1 Moisture content of starch

As discussed in section [2.3], the properties of starch change with moisture content. Ollett *et al.* [2] and Kirby *et al.* [62] have shown that the moisture content of starch varies with a change in relative humidity of the environment. In their papers they did not publish the relative humidities of the environment, but stated which saturated salt baths were used to control humidity. The following work uses data from Ollett *et al.* [*ibid*] to obtain a relationship between relative humidity and moisture content of starch. This relationship is then used with further data from Ollett *et al.* [*ibid*] to obtain a relationship between relative humidity and the elastic modulus of starch.

Table [3.2.1] – Moisture content and modulus of elasticity of starch for a range of relative humidities of the environment controlled by saturated salt solutions [2].

Saturated Salt Solution	Relative Humidity (%)	Moisture content (% wt)	Elastic Modulus MPa
Magnesium Chloride	33	9	4500
Potassium Acetate	23	7	5000
Potassium Carbonate	43	10	4000
Ammonium Sulphate	81	17	1900

Table [3.2.1] shows the resultant relative humidities generated for the corresponding saturated salt baths [2]. It also shows the moisture content (as a percentage of the weight) and elastic modulus for the salt solutions as measured by Ollett *et al.* [*ibid*] and Kirby *et al.* [*ibid*].

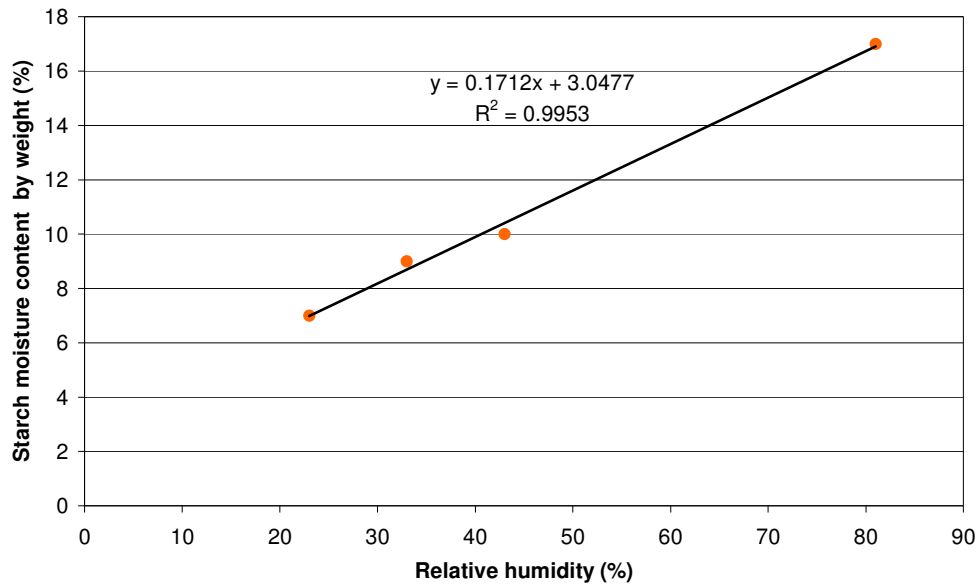


Figure [3.2.1] – Relationship between relative humidity of an environment and starch moisture content.

The relationship between relative humidity (x) and starch moisture content by weight (y) is illustrated in figure [3.2.1]. The relationship seen is linear with a correlation figure of R^2 of greater than 0.99 and a line of best fit given by $y = 0.17x + 3.0$.

3.2.2 The effect of moisture upon the elastic modulus of starch

The relationship between relative humidity (x) and elastic modulus (y) is illustrated in figure [3.2.2]. The relationship is again highly linear with a R^2 value greater than 0.99 and a least squares line of best fit given by $y = -53.7x + 6370$.

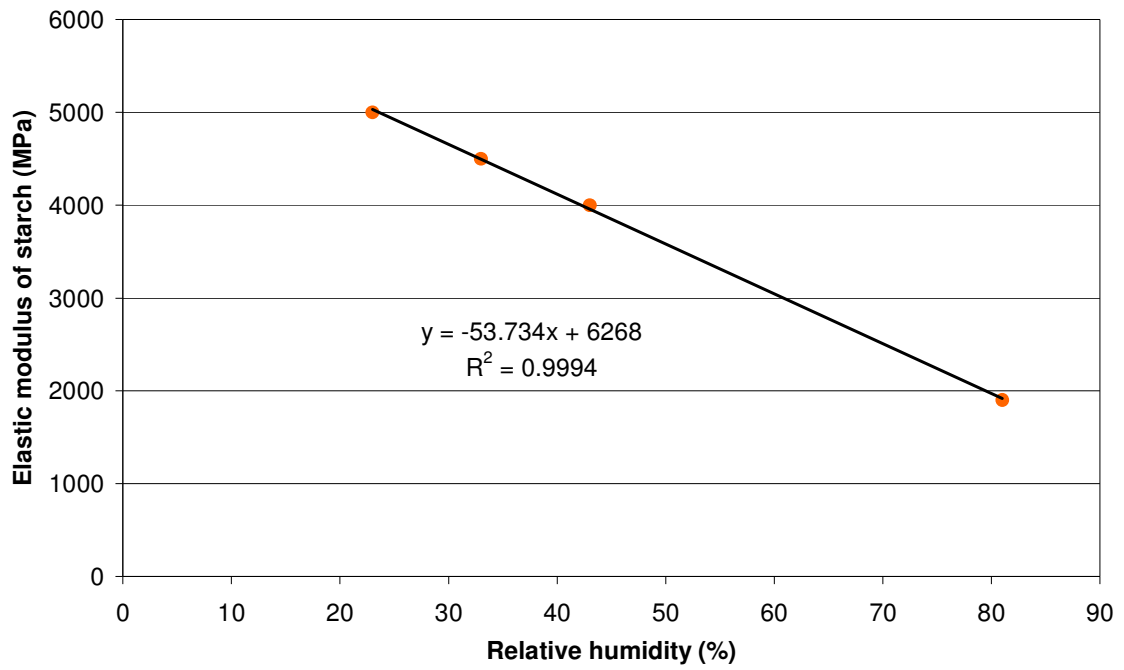


Figure [3.2.2] – Relationship between relative humidity of an environment and modulus of elasticity of starch.

These relationships are used in the finite element analysis as discussed in section [3.4.2]

3.3 Calculation of the average strain from washboarding profiles

Strain is defined as the ratio of the change of length to the initial length. The average strain, S , due to glue forces may be calculated from an accurately measured washboarding profile, where figure [3.3.1] illustrates the geometry of the components for calculating the strain. A washboarding profile is made up of a number of segments, where the distance separating each point in the profile is defined as the segment length, Δx . The strain may differ for each segment so that the strain (ϵ) can be considered to be an approximation to the average strain for all the segments within the flute pitch L_{wb} .

The length, L_{wb} is equal to the distance between the peaks in a washboard measurement profile and each point of the profile is displaced a distance, ΔD_{wb} , vertically from the previous point.

It was assumed for the calculations in this section and section [3.3], that L_{wb} remains the same for a change in washboarding depth. Fourier analysis was used to check if this assumption is valid by transforming washboarding profiles with different depths to see if the frequency (corresponding to a change in washboarding pitch) changes as the washboarding depth changes. If it doesn't then we may conclude that the pitch, L_{wb} , remains the same. The results presented in section [6.2.2] confirm this assumption.

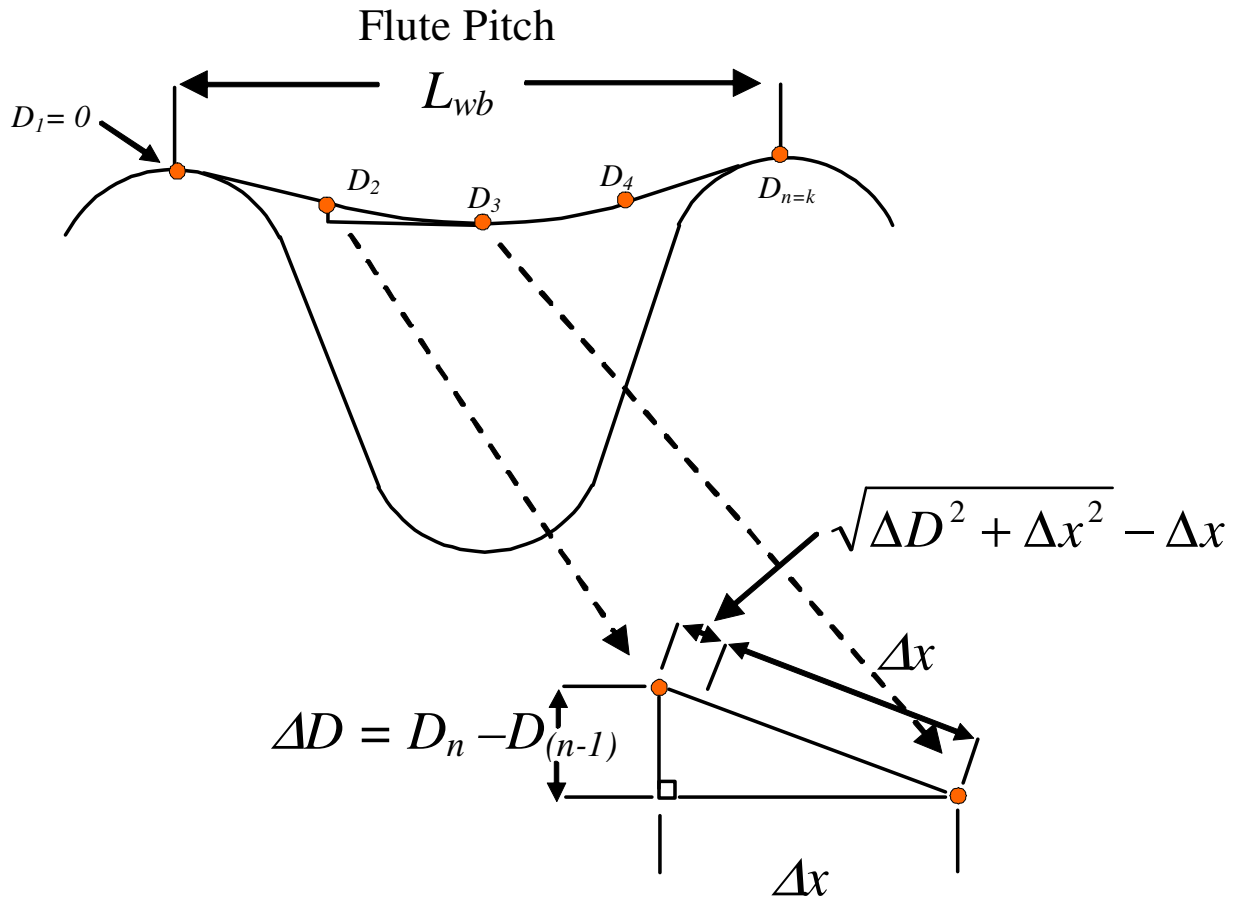


Figure [3.3.1] – Geometry of strain measurement from a washboarding profile.

The strain may then be calculated using,

$$\varepsilon = \frac{\sum_{n=1}^{n=k} \left[\sqrt{(D_n - D_{n-1})^2 + \Delta x^2} - \Delta x \right]}{L_{wb}} \quad \text{Equation [3.3.1],}$$

where ε is the average strain, L_{wb} is the flute pitch, $n = 1$ is the first sample and $n = k$ the final sample.

3.4 Finite element analysis (FEA) of washboarding

3.4.1 Finite element analysis overview

FEA is an indispensable method used for modelling or simulating complex real world material or mechanical systems. It is most useful for non-linear highly complex systems that are impossible or very difficult to solve by analytical analysis. FEA is a discrete method that uses information such as mathematical equations describing real world physical behaviour, look-up tables (e.g. for non-linear material properties), boundary conditions, contact detection of analytical bodies and matrix arithmetic to model real world systems.

FEA involves building an accurate two or three-dimensional geometric model of the system to be analysed for its response to external stimuli. This model is divided into discrete elements with nodes at their corners. Material properties are assigned to these elements, generally in the form of stress and plastic strain curves with the initial elastic modulus and yield load which, for the case of paper, is known to be highly non-linear.

The structure is broken down into many small simple blocks or elements for finite element analysis. The behaviour of an individual element can be described with a relatively simple set of equations where the most fundamental equation is Hooke's law, $F = kx$, where F is force on an element, k is the spring constant of the element and x is the elongation of the element.

Just as the set of elements would be joined together to build the whole structure, the equations describing the behaviour of the individual elements are joined into an extremely large set of equations that describe the behaviour of the whole structure. The computer can solve this large set of simultaneous equations by using algorithms based on matrix algebra. From the solution, the computer extracts the behaviour of the individual elements and from this it can then determine the stresses and deflections of all the elements or parts of a structure.

The term "finite element" distinguishes the technique from the use of infinitesimal "differential elements" used in calculus, differential equations, and partial differential equations. Finite element analysis is a way to deal with structures that are more complex than can be readily dealt with analytically using partial differential equations

Boundary conditions are used to model physical connections by defining the appropriate degrees of freedom (DOF) for all boundary nodes. Each node has up to six DOF with three assigned to translation and the other three for rotation. Appropriate boundary conditions can be used for modelling dynamic, thermal, acoustic, fluidic and electrostatic conditions.

Analytical contact bodies may be defined to simulate bodies that in the real world come in contact with the main body under analysis, for example, a platen compressing a corrugated board sample. A contact body movement can be translated or rotated by defining angular or linear displacement, velocity or acceleration.

After the geometry, material properties, boundary conditions, and contact bodies have been defined, the next step is to decide what element type(s) will be used. The elements are then meshed onto the geometry using element sizes appropriate for their location and curvature of surfaces. Areas where high stress concentrations or gradients might exist will have smaller elements assigned.

The FEA solver is then run to solve thousands to hundreds of thousands of simultaneous linear and non-linear differential equations to obtain a physical displacement of each node. This displacement data is then used to calculate the force at each node, strain in the element and stress gradients in the element, each of which can be mapped as colours onto the

undeformed or deformed geometry.

3.4.2 Emulation of glue quantity in finite element analysis of washboarding.

The relationship between the elastic modulus of starch and the relative humidity was used in the finite element analysis studies to estimate the change in simulated glue force between the liner and fluting, corresponding to a change in relative humidity (see table [3.4.1]).

Table [3.4.1] – Modulus of elasticity of glue for a range of relative humidities of the environment and relative simulated glue force applied in finite element analysis.

****Simulated glue force is relative to the force applied by the starch at 50% relative humidity.***

Relative Humidity (%)	Elastic Modulus (MPa)	Simulated Glue Force*
23	5032	100
50	3581	71
78	2077	41
90	1432	28

The simulated glue force values in this table are relative to the force applied in an environment of 23 % relative humidity, where the modulus of elasticity of the starch was approximately 5032 MPa.

Starch once set (or equilibrated in moisture content) at a given humidity has a specific modulus of elasticity and hence stiffness. A change in stiffness, by definition, changes how much starch resists deformation. A change in this resistance to deformation is hypothesized to coincide with a resultant proportional change of glue force upon the liner.

The FEA performed in this thesis (see section [3.4]) is a static analysis and hence the glue is assumed to have lost most of its moisture to a point where it has set, and is modelled in this way. Dynamic modelling of the process of shrinkage was beyond the scope of the work presented in this thesis but could be considered for future work if information about rates of shrinkage and changes in glue modulus at the relevant time scales can be obtained.

For these reasons, the shrinking of the glue was assumed to have occurred and any change in modulus was simulated in the FEA as a change in the force applied to the liner, varied according to an appropriate modulus of elasticity for a given relative humidity.

The following formulates the relationship between the simulated glue force F_{RH} at a given relative humidity and the glue force F_{23} corresponding to the amount of simulated glue force that provided an amount of washboarding depth (50 μm) at a relative humidity of 23 %. The value of 50 μm is typical washboarding depth as encountered in the physical testing (see table [6.4.1] and figure [6.4.1]). This force was then designated as a 100 % simulated glue force.

The relationship, using the definition for elastic modulus for a relative humidity of 23 %, may be defined as

$$F_{23} = A_g \cdot E_{23} \cdot \varepsilon \quad \text{Equation [3.4.1],}$$

where F_{23} (N) is the internal force of the glue at 23 % RH, A_g (m^2), is the cross sectional area of the glue, E_{23} (N/m^2), is the elastic modulus of the glue at 23 % RH and ε is the strain experienced by the glue.

Similarly,

$$F_{RH} = A_g \cdot E_{RH} \cdot \varepsilon \quad \text{Equation [3.4.2],}$$

where F_{RH} (N) and E_{RH} (N/m^2), are respectively the internal force of the glue and the elastic modulus of the glue at a particular relative humidity.

Hence the forces to simulate a change in elastic modulus were calculated using

$$F_{RH} = \frac{E_{RH}}{E_{23}} \cdot F_{23} \quad \text{Equation [3.4.3],}$$

and are shown in table [3.4.1].

It should be also noted that at no stage are changes in paper stiffness ignored and changes in stiffness are included in the methodology of the FEA modelling presented in section [3.4].

3.5 Measurement of full-tone print coverage

In this section the theory underlying the algorithms used to quantitatively measure the amount of full-tone striping are presented.

A greyscale digital image of the striped region on the board under question was acquired and subsequent analysis involved segmenting the data from three regions of the image. One segment corresponded to the background board colour (or base intensity) I_{jk} . The second segmented region was an area, P_{jk} , defining the ink colour (or intensity) and the last was the total printed area, A_{jk} , to be analysed, where the printed area under investigation included an equal amount of washboarding valley tops and bottoms. Figure [3.5.1] shows such an area. The image pictured in figure [3.5.2] is not used for printing analysis but is included for the reader's benefit. It shows the image in figure [3.5.1] altered for maximum contrast.

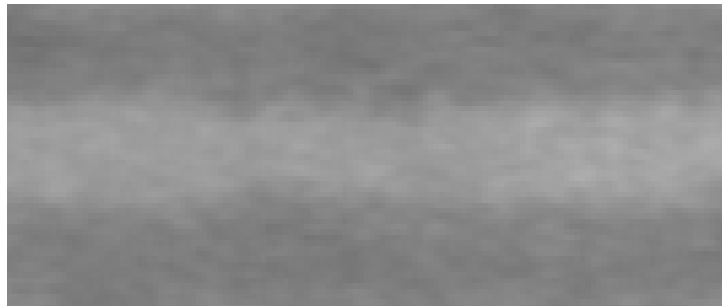


Figure [3.5.1] – The area, A_{jk} , used for print coverage analysis.

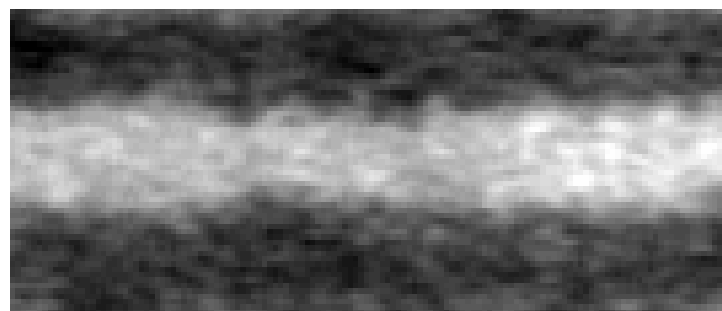


Figure [3.5.2] – The image in figure [3.5.1] increased in contrast for improved visual inspection (not used in analysis).

The mean of all values in I_{jk} , (the base intensity), was defined as B_I ,

$$B_I = \sum_{j=1}^{n_{bw}} \sum_{k=1}^{n_{bh}} \frac{I_{jk}}{n_{bw} \cdot n_{bh}} \quad \text{Equation [3.5.1]}$$

where n_{bw} and n_{bh} were the number of data points defined as the widths and heights respectively of the base area.

Similarly, the mean of all values in P_{jk} , (the ink intensity), was defined as D_P ,

$$D_P = \sum_{j=1}^{n_{pw}} \sum_{k=1}^{n_{ph}} \frac{P_{jk}}{n_{pw} \cdot n_{ph}} \quad \text{Equation [3.5.2]}$$

where n_{pw} and n_{ph} were the number of data points defined as the widths and heights respectively of the ink coverage area.

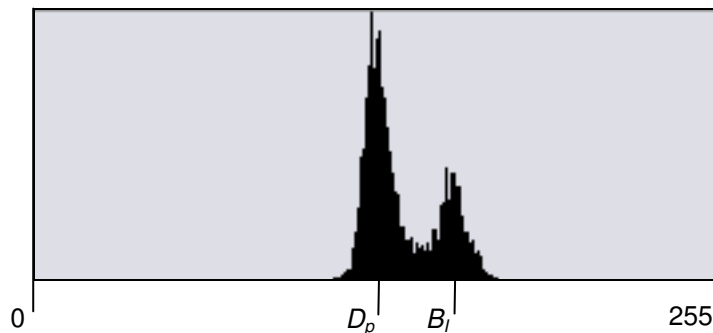


Figure [3.5.3] – Intensity histogram of the values in the area, A_{jk} , as illustrated in image [3.5.1]. It shows the values B_I and D_P visually

Figure [3.5.3] above illustrates the values B_I and D_P visually on a histogram of intensity values in the area, A_{jk} , as shown in figure [3.5.1]

The nature of full-tone printing is binary ie. a given small area on the surface of the board in question may have coverage of full tone print or not. Simple averaging would have led to uncertainties in defining coverage, as external effects such as non uniformity of lighting conditions would contribute to the measurement.

Therefore the area under investigation was binary thresh-holded to two values, 0 (black in

greyscale) and 255 (white in greyscale), with all points with values below the intensity at the midpoint between B_I and D_P as having no coverage (black) and those intensities greater than or equal to the midpoint as having coverage (white).

$$A_{jk} \geq \left(\frac{B_I + D_P}{2} \right) \Rightarrow A_{jk} = 255 \quad (\text{white in greyscale}) \quad \textbf{Rule [3.5.3]}$$

$$A_{jk} < \left(\frac{B_I + D_P}{2} \right) \Rightarrow A_{jk} = 0 \quad (\text{black in greyscale}) \quad \textbf{Rule [3.5.4]}$$

Figure [3.5.4] below shows the image in figure [3.5.1] after binary thresh-holding.



Figure [3.5.4] – The area, A_{jk} , after thresh-holding

The mean (T_A) of the values of the total area in A_{jk} was then calculated using the following equation,

$$T_A = \sum_{j=1}^{n_{tw}} \sum_{k=1}^{n_{th}} \frac{A_{jk}}{n_{tw} \cdot n_{th}} \quad \textbf{Equation [3.5.5]},$$

where n_{tw} and n_{th} were the number of data points defined as the widths and heights respectively of the area under investigation.

The value T_A is then converted to the print ratio using the following equation,

$$PR = \left(1 - T_A \cdot \frac{1}{255} \right) \cdot 100\% \quad \textbf{Equation [3.5.6]}.$$

The print ratio, PR , in equation [3.5.6], corresponds to values of print coverage in the range of

0 to 100%, where higher values of PR equate to more coverage and less striping and vice versa for lower values.

Chapter 4 – Methods Part A – Measuring Washboarding

4.1 Digital image profilometry - Overview

A reliable bench top instrument was designed and constructed in conjunction with software so that a one dimensional profile of the corrugated cardboard surface could be measured. The instrument was based on a design by Reich and Allan [4] and investigated by Gomez [5]. A summary of the differences is presented at the end of this section. The hardware captured images that were analysed by the software to extract the washboarding profile.

Measurements of the washboarding severity and geometry can be made using this system within seconds. Without the use of this hardware and software it would have been very difficult to gain this information in a reasonable time frame. Had this been done in the traditional way of scanning profilometry, it would take months not days to analyse the large number of images acquired during this study.

4.1.1 Washboarding Hardware

The configuration of the profilometer is shown in figure [4.1.1] and consisted of:

- A 3 mW laser diode light source operating at 680 nm (red);
- A grating of opaque and clear parallel lines with a period of 200 μm on a glass slide;
- An image acquisition board (see section [4.4.1(a)]);
- A charged coupled device (CCD) camera mounted on an adjustable arm (see section [4.1.1(b)]);
- A pneumatic corrugated board sample holder (see section [4.1.1(c)]);
- A light tight enclosure with doors for access to the sample holder (not pictured in figure [4.1.1] but can be seen being used in figure [4.2.4])

- And a, 75 MHz Pentium personal computer, running a multitasking operating system (MS Windows 95).

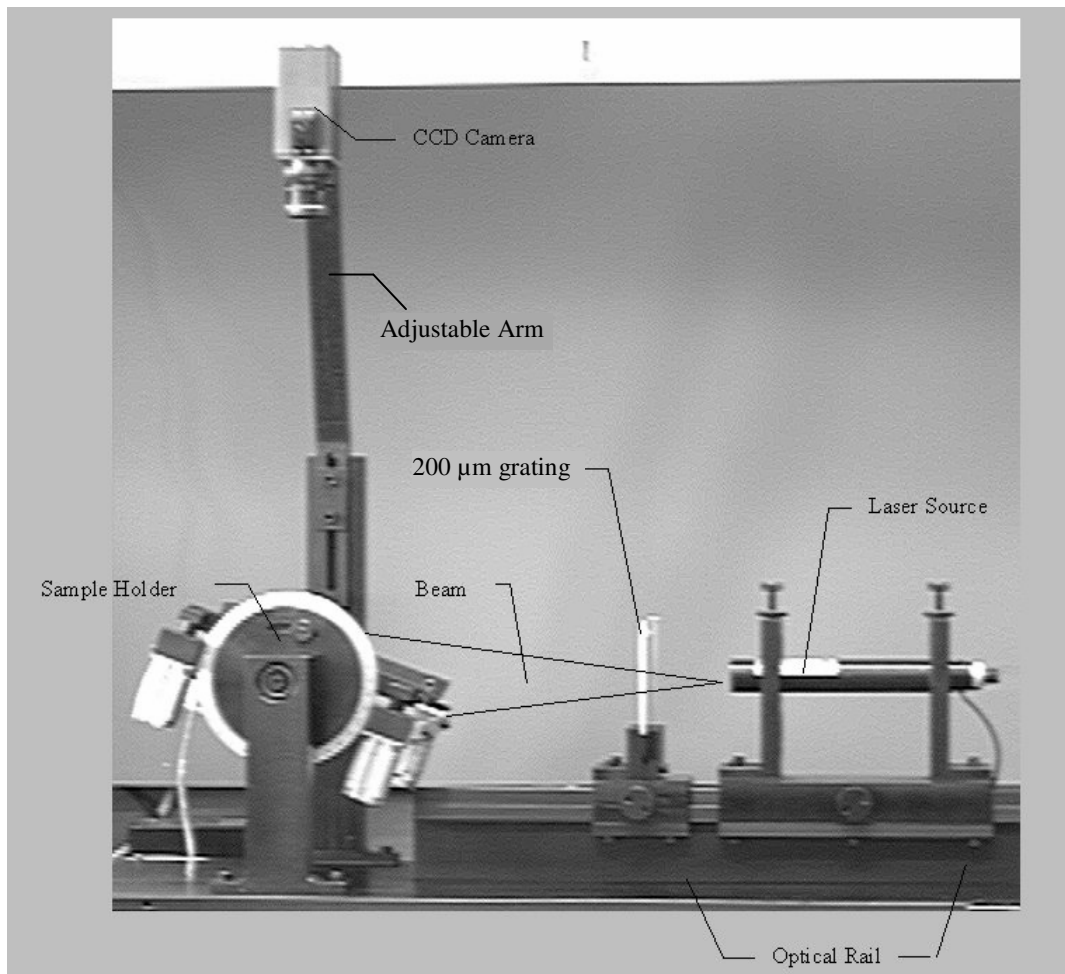


Figure [4.1.1] – Washboard measurement set-up

4.1.1(a) Image acquisition board.

A Data Translation (model DT2853-50Hz), monochrome image acquisition board capable of capturing 512 x 512 x 8 bit video images in real time was used. It was supplied with MS Windows dynamic link libraries (DLL) to enable capturing of images in custom written software such as the washboarding software (see section [4.1.2] below) written by the author. These libraries were used within the main program to capture the images for washboarding analysis and either analysed immediately or saved to disk for later analysis.

4.1.1(b) Charged coupled device (CCD) camera

The CCD sensor array produced greyscale images. These images had a dynamic range of 8 bits before being converted to an analogue composite PAL signal (768 pixels by 578 interlaced lines). The image acquisition board converted the analogue signal back to digital later (see section [4.1.1(a)]). The lens was interchangeable, however only one lens was used with a fixed focal length of 50 mm. It had an adjustable aperture to optimize the use of available light and to prevent saturation of the image sensor array. The camera was mounted such that the camera to sample distance was adjustable, enabling modification of the area captured by the camera.

4.1.1(c) Pneumatic sample holders

Corrugated board samples (150 mm wide by 200 mm long) were held in place by the use of four pneumatic actuators which pulled down two bars (5 mm wide by 200 mm long) on either side of a corrugated sample when air pressure was applied. Damage to any sample was minimised by adjusting air to the actuators, decreasing the force on the board required to hold a sample. The bars also had a padding of soft foam rubber, further minimising any damage to the corrugated sample. The image width was typically around 80 mm compared to the width of the corrugated samples of 150 mm and therefore the bars holding the sample were approximately 30 mm distant from the area under investigation on a sample. With these precautions adverse effects on the measurement of washboarding by the sample holders were minimised.

4.1.2 Washboarding software

4.1.2(a) Extraction of washboarding profile software

The computer software used in conjunction with the profilometer was written and developed as part of the work for this thesis by the author. The program ran in a 32-bit Windows™

operating system and was further developed as new measurements were devised. These developments included measurement of the extent of full tone print coverage, general improvement in the measurement technique of washboarding profiles and shape analysis of the resulting washboard profile. For a summary of improvements see section [4.3] at the end of this chapter.

The software used for measuring the washboarding profiles was written using Borland C++ version 5 and was developed to run on the same personal computer which included the image acquisition board. The source code for this program is attached in Appendix A. The methods used by this software to extract a profile are presented in the following sections.

4.1.2(b) Extraction of washboarding shape and strain software

A second piece of software was written by the author to calculate the strain (see section [3.3]) and extract the shape (see section [5.3.3]) of washboarding. The software was written using Borland C++ Builder 3, of which the source code of the program is presented in Appendix B.

4.2 Measurement of Washboarding Profiles

4.2.1 Projected lines

The light emitted from the laser diode was projected through the grating onto the corrugated cardboard sample, forming a series of approximately parallel lines. The series of lines followed the undulating surface profile of the sample board. The angle of incidence of the light and therefore the line separation could be varied to suit by rotating the sample holder in a range of 0 - 30 degrees to the horizontal. Both the grating and laser were mounted on an optical rail for robustness and easy adjustment.

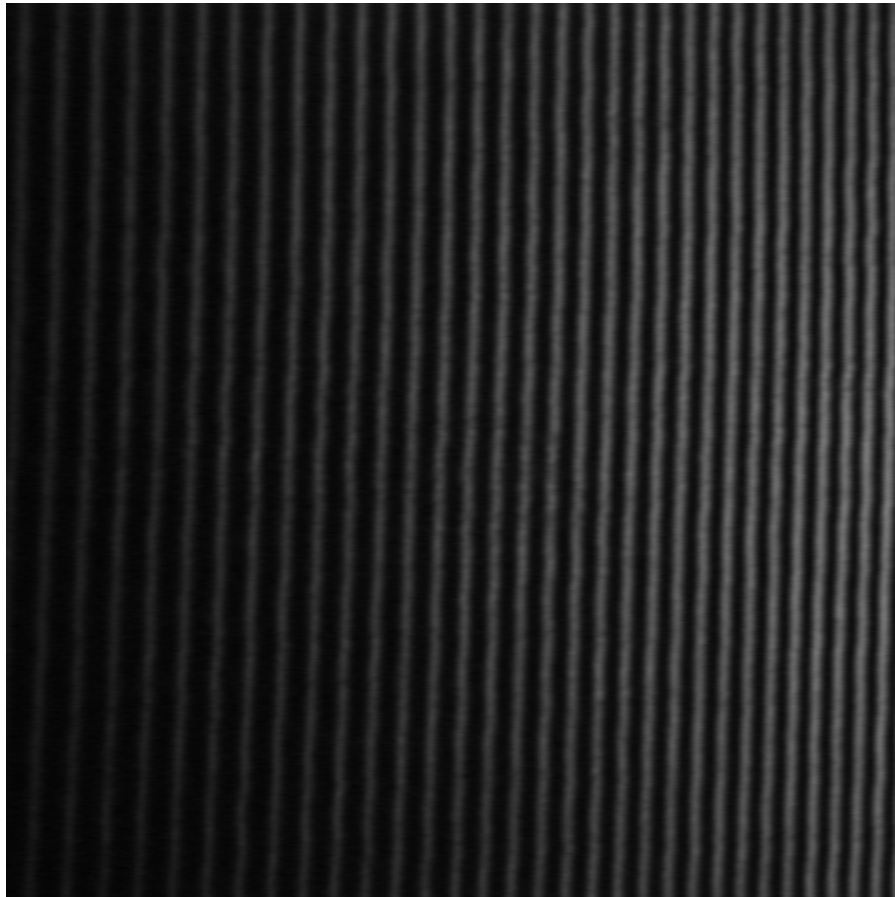


Figure [4.2.1] – An image captured for the purpose of measurement of washboarding. It represents an area of approximately 8 x 8 cm (64 cm²)

The CCD camera captured multiple images of the series of lines projected onto the sample. These images were averaged to reduce noise in the final image used for washboarding profile extraction. The signal to noise ratio increased by 3dB for each doubling of the number of

images averaged. By capturing 32 images, the noise was reduced by approximately 15 dB [82, 84]. If necessary, the camera's lens could be changed and the camera-to-sample distance varied. The image acquisition board (Data Translation – DT2853) acquired a grey scale image of 256 greyscale levels of size 512 x 512 pixels. An example image is shown in figure [4.2.1], representing an area of approximately 8 x 8 cm (64 cm²).

The lines that were projected onto the corrugated cardboard were approximately periodic and approximately sinusoidal due to the penumbral effect. The penumbral effect is the creation of partial shadows at the edges between regions of full shade and full illumination. If the projected fringes were not approximately sinusoidal then the corresponding spectrum through Fourier transformation would contain harmonics, complicating the extraction of a phase profile.

4.2.2 The method of extracting a profile

The projected lines, as described in the previous section, were distorted by the surface profile of the corrugated cardboard and this distortion was used to extract a surface profile. Figure [4.2.2] depicts a flow chart describing an overview of the method of extracting a profile described within this section up to, but not including, filtering and calibration.

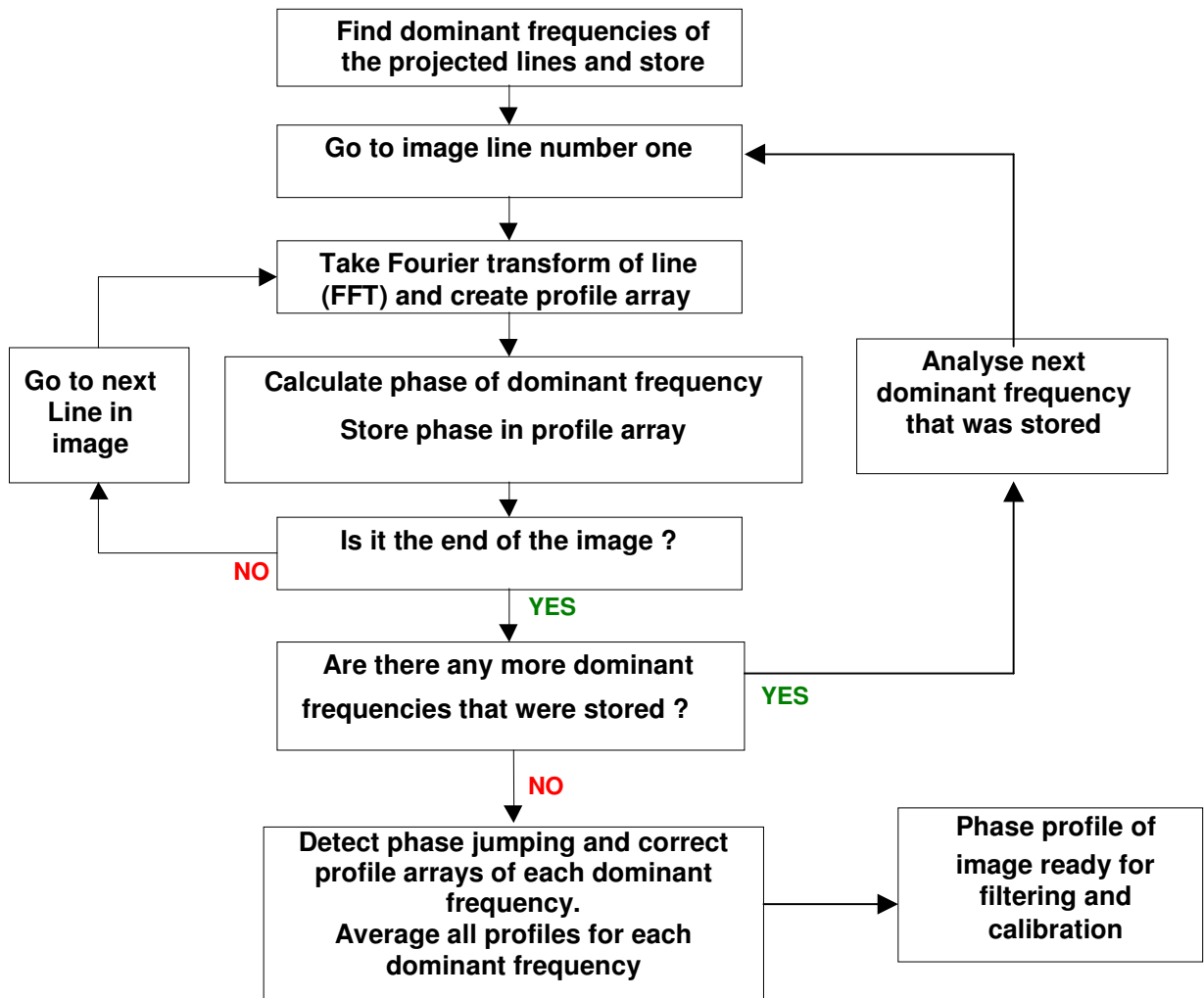


Figure [4.2.2] – Flow chart describing the method of profile extraction from an image of projected lines.

The left-hand side of figure [4.2.3] shows the frequency power spectra of each horizontal image line shown on the right hand side (see section [3.1] for Fourier analysis theory). Figure [4.2.4] shows, in more detail, the frequency spectrum of a single horizontal line.

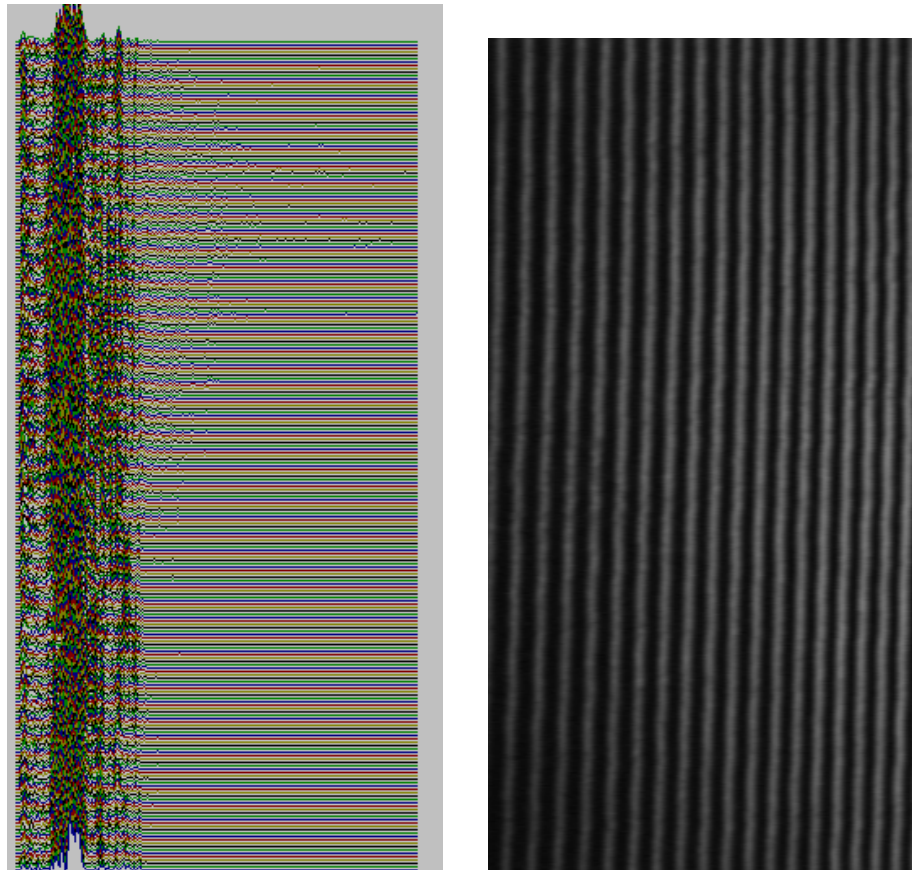


Figure [4.2.3] – The frequency spectrum (left) is constructed from the Fourier Transformation of projected fringe lines (right).

A limited range of frequencies (centred on the fundamental ‘dominant’ frequency), were used in the analysis, reducing residual information unrelated to the projected fringes contained within the frequency spectrum. The range of frequencies was set by applying a threshold to eliminate frequency components whose amplitudes were below a specific percentage of the amplitude of the dominant frequency.

These lower amplitude frequencies may have been due to variations in illumination, non-uniformity in the camera response and random noise as a result of distortion of the fringes due to surface roughness and were not used in the extraction of phase profile by means of the thresholding as described previously.

In order to maximize the number of frequency components used in the analysis, the threshold percentage at which a frequency component was included in the analysis was made to be a user adjustable parameter. This was adjustable because the amplitudes of the frequency components differed for the different flute types (e.g. B and C-flute).

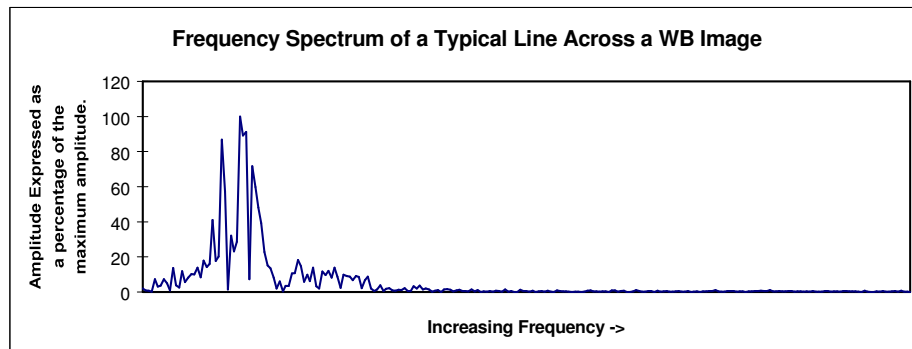


Figure [4.2.4] – A typical frequency spectrum across a horizontal line of an image.

To extract how the surface varies in height, the phases of the dominant frequencies (ie. frequencies greater than a selected minimum amplitude threshold) were used. The phase was calculated for all dominant frequencies for each horizontal image line.

The combination of all the phases from the first (top) horizontal line to the last (bottom) horizontal line of an image of a frequency were combined to achieve a single phase profile. This was repeated for each frequency above the threshold. The final profile was then an average of each individual profile.

The phase of a frequency was calculated by using the real and imaginary parts of the Fourier transform for the final profile.

The formula used to calculate the phase was

$$\theta = \tan^{-1}\left(\frac{\text{imag}}{\text{real}}\right) \quad \text{Equation [4.1.1]}$$

Using equation [4.1.1] in conjunction with the signs of the real and imaginary parts, a phase profile in the range from 0 to 2π was constructed. Figure [4.2.5] shows a typical profile.

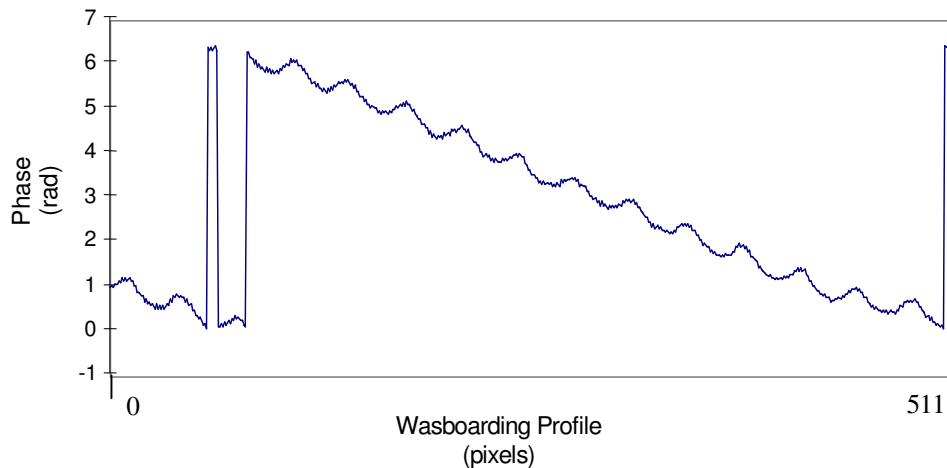


Figure [4.2.5] – Profile before phase unwrapping

4.2.3 Phase unwrapping

To extend the phase range to beyond 2π , the phase profile was unwrapped. Phase unwrapping consisted of the detection of jumps in phase and subsequent adjustment by adding or subtracting multiples of 2π . Figure [4.2.6] shows the data after phase unwrapping.

The slow linear phase change seen in figure [4.2.6] is largely due to misalignment of the grating. The slight curvature superimposed on the linear phase change may be due to the slight bending of the corrugated cardboard sample or possible optical distortion such as curvature of the field.

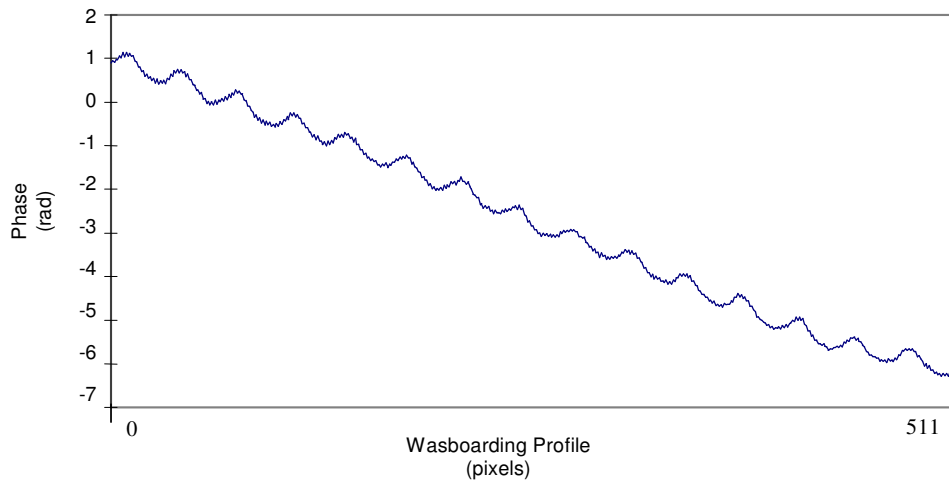


Figure [4.2.6] – Phase unwrapped profile

4.2.4 Filtering

What appears as high frequency noise on the phase profile may either be quantisation noise or due to protruding fibres on the sample’s surface. Primarily, only the lower frequency washboarding content was required. The reduction of low frequency bending and high frequency roughness due to fibre protrusion and quantisation noise was achieved by using an adjustable band-pass recursive digital filter.

Figure [4.2.7] shows the frequency response of a typical filter and figure [4.2.8] illustrates the resultant phase after using this filter. The filter could be altered to suit the type of profile under investigation. For example A-flute washboarding was of a different period compared to C-flute.

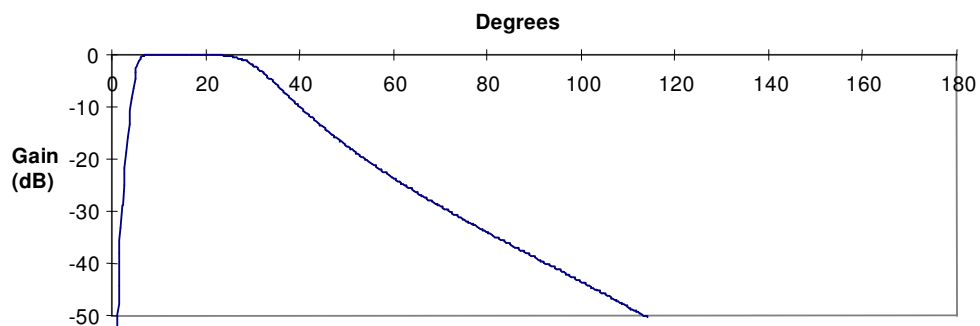


Figure [4.2.7] – Frequency response of recursive digital filter

4.2.5 Calibration

A method was developed to calibrate the instrument and software and is described next. At this stage of the analysis, the washboarding profile depth was expressed as phase information in radians, as shown in figure [4.2.8]. Conversion from phase information to absolute distances was achieved by performing a similar analysis with a flat steel plate in place of a corrugated board.

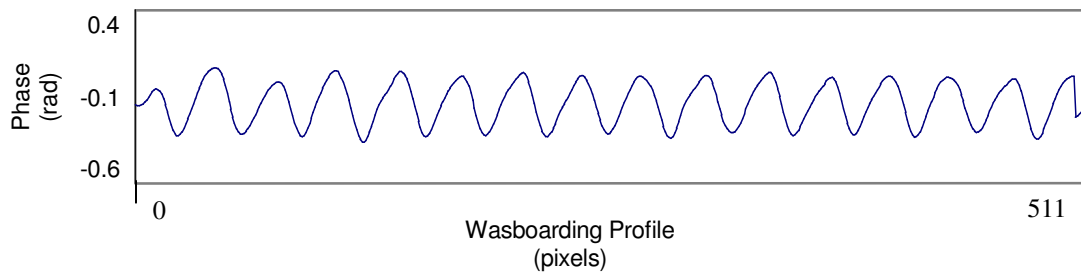


Figure [4.2.8] – Surface profile after filtering

Figure [4.2.9] shows a steel plate in place in the profilometer for the purpose of the calibration and figure [4.2.10] shows the image capture of the steel plate. The profilometer was used to extract the profile of this plate at an angle, however, the profile, unlike for the washboarding profiling, was left unfiltered. This resulted in a phase profile such as seen in figure [4.2.11]. From a linear best fit to the phase and knowledge of the physical dimension of the plate and the angle of the plate with respect to the horizontal, the phase was calibrated in terms of μm per radian.

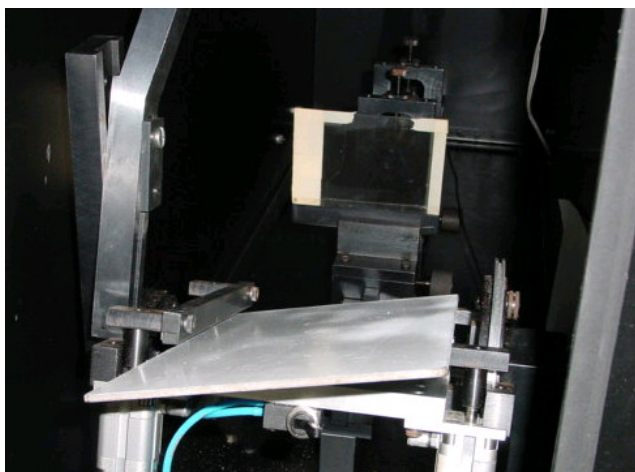


Figure [4.2.9] – Profilometer with calibration steel plate in place.



Figure [4.2.10] – Acquired image of steel plate used for calibration.

Multiplication of each point in the phase profile of corrugated cardboard by the calibration factor resulted in absolute dimensions of the washboard profile for the vertical ordinate shown in figure [4.2.12].

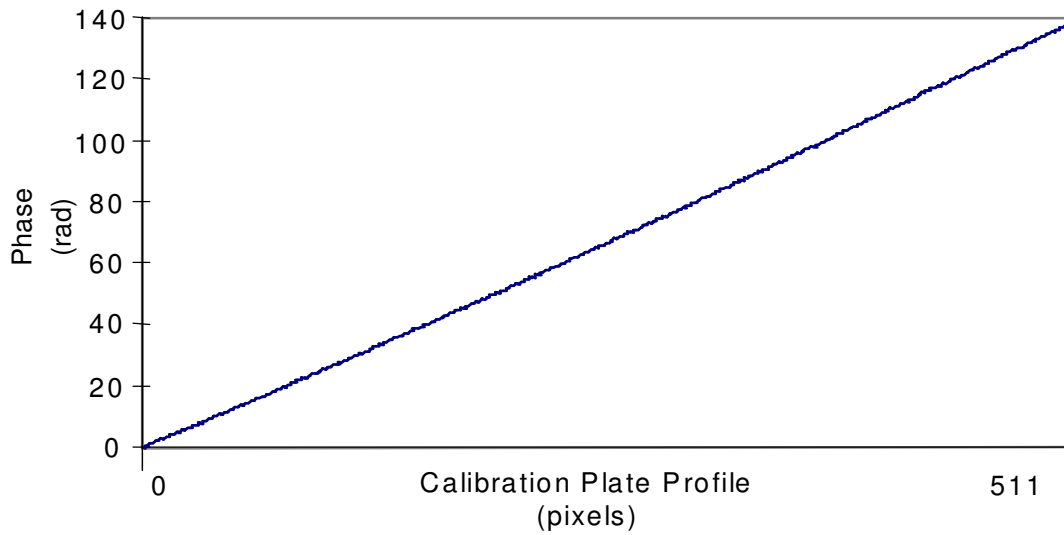


Figure [4.2.11] – Resultant phase profile of image in figure [4.2.10]

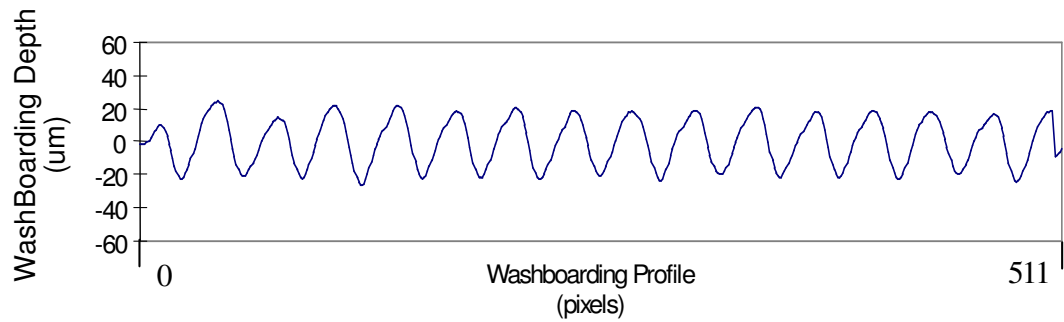


Figure [4.2.12] – Washboard depth in units of distance and pixels.

The horizontal scale of the phase profile was calibrated by capturing an image of a one millimetre ruled grid in the apparatus. Figure [4.2.13] shows such a captured image.

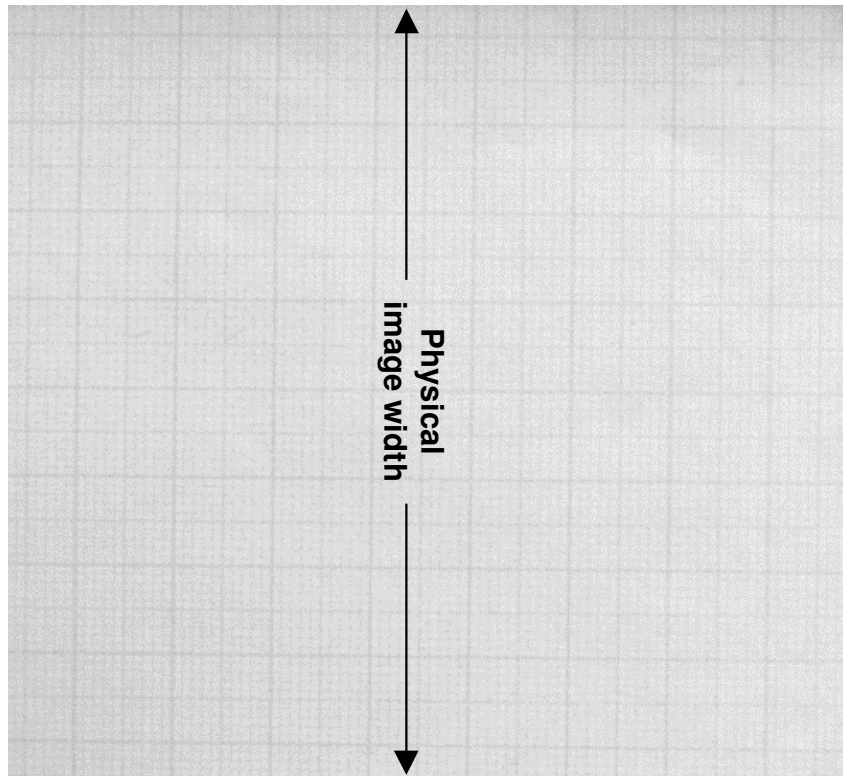
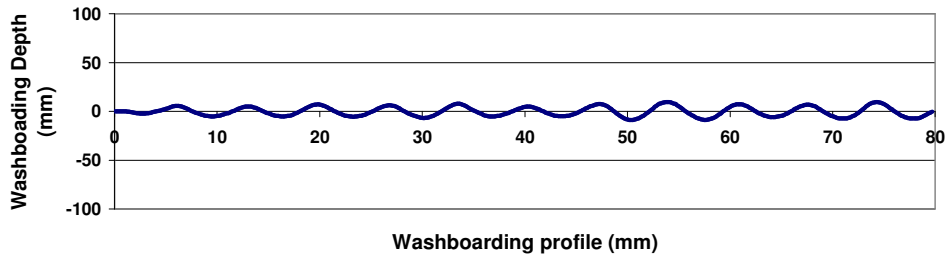


Figure [4.2.13] – Image for abscissa calibration

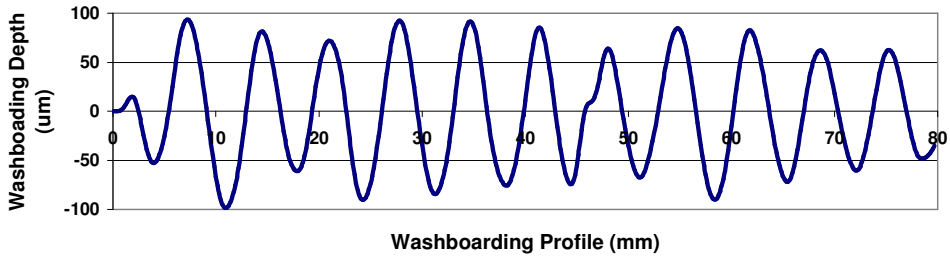
A conversion factor of millimetres per pixel was calculated using the physical image width obtained by extraction from the measured number of pixels corresponding to the image width. For example in figure [4.2.13], the physical image width is 79.5 mm and the width in pixels is 512 pixels. The conversion factor for these values was therefore 6.44 pixels/mm.

4.2.6 Final washboarding profile

Figure [4.2.14] shows two examples of different washboard severity, one with ‘minimal’ washboarding and the other ‘extensive’.



(a) *Minimal*



(b) *Extensive*

Figure [4.2.14] – Two profiles showing different severities of washboarding.

4.3 Original contributions to the profilometer technique.

The following is a summary of the author's contribution to the original profilometer technique developed by Reich and Allan [4]; software written by Reich and as used by Gomez [5]

- Improvements to the profilometry technique:
 - The original technique called for a single image to be analysed. The signal to noise ratio was increased by capturing more images and averaging them.
 - The original methodology incorporated the fitting of a polynomial of n degrees to the measured profile to reduce the misalignment of the field and/or curl of a corrugated sample. This often did not work well in the presence of noise. It was replaced by a band-pass recursive filter, which not only was more robust but also had the added benefit of reducing high frequency noise from a measured profile.
 - A reliable calibration method of the instrument was developed to produce a profile with physical dimensions. Previously the profiles were not calibrated.
 - The original technique encompassed measuring the phase profile of the single most dominant frequency in the image. The distance between the projected lines on the sample as captured by the camera was not constant. Therefore there was no single frequency that represented all of the lines on the sample and hence the technique was improved by averaging the profiles of the range of frequencies representing all of the projected lines.

- Design of a stable and robust bench top instrument:
 - The original apparatus was inflexible as it had little range of adjustments to optimize distances between parts. This situation was improved by the design of a new bench top instrument with an optical rail and an adjustable camera arm.

- A pneumatic holder of samples were added which reduced the curl of a corrugated sample. When measuring washboarding depths at a range of 10 to 150 μm , even a slight curvature of the board can dominate the profile to such an extent that filtering or fitting a polynomial to the profile would introduce unacceptable uncertainties about the quality of a measured profile.
- The original equipment incorporated a laser light source that had a lens to focus the laser light into a parallel beam which was then passed through an objective lens to spread the parallel beam through the grating onto a sample to be profiled. Both lenses were removed. This removed unnecessary complexity as the technique called for a spreading of the beam. It also increased light intensity considerably, reducing noise from system.
- Software Development:
 - The technique was extended by incorporating an automated method of capturing multiple images at well defined intervals. This was very useful for measuring washboarding within an environmental room of cyclic humidity.
 - The original method was a two step process of capturing the images in one program and analysing with a second. The method was optimised by incorporating image capture and analysis into the one piece of software. This enabled a quick re-capture if the original image turned out to be of poor standard.
 - The original technique used a TV monitor to align the parts (camera, grid and laser) and to adjust the aperture and focus of the lens. This was not necessarily what the image capture card would acquire. This situation was improved by live streaming of video from the camera into the main program.

Chapter 5 – Methods Part B – Investigations

5.1 The effect of starch (glue) quantity upon washboarding

5.1.1 Washboard depth and glue coverage – Manually manufactured board

The measurement of washboarding depth as a function of glue quantity was performed to determine how varying the glue quantity alters the washboarding. The method used to extract this information is described below.

Materials used included wheat starch (glue), standard C-flute single facer corrugated boards and liners of composition and basis weights as shown in table [5.1.1].

Table [5.1.1] – Basis weight and compositions of papers used in glue quantity investigation - Manually manufactured

Paper	Grammage (gsm)	Composition
single facer	205	Kraft Liner Board: mixture virgin softwood (70 %) and virgin hardwood eucalypt (30 %)
fluting	150	Recycled waste
double backer	210	Kraft Liner Board: mixture virgin softwood (70 %) and virgin hardwood eucalypt (30 %)

The apparatus used included an electronic scale accurate to 0.001 g, a hot plate, a glass plate and steel starch spreader, a 2 kg weight and the washboard profilometer.

Sample boards with quantities of applied glue were manufactured by hand. Starch was poured onto a glass plate where a thin film of glue was created by the use of metal spreader with a small gap. The method is illustrated in figure [5.1.1]

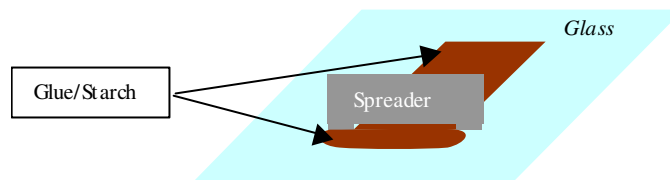


Figure [5.1.1] – Method of controlling glue application

Running the glue spreader at various angles to vertical allowed the glue gap to be altered and therefore the quantity of glue used could be changed.

Once the glue was spread on the glass surface, the single facer was laid down on the glue with the flute side facing downwards. The single facer was then removed from the surface and a liner previously cut to size placed on the single facer, forming the sample to be measured for washboarding. The board was then placed on a hot plate at approximately 200 °C with a 2 kg weight placed onto it for 10 seconds, which emulated the process used on a corrugator. The liner and single facer was weighed prior to the assembly of the three components. The samples were weighed after allowing the moisture in the paper to reach equilibrium in ISO conditions for one week (see section [5.2.1]). Knowledge of the weight of the liner and single facer before and after the process allowed the calculation of the amount of glue applied.

The quantity of glue applied equals the weight of the sample after assembly minus the combined weight of the liner and single facer prior to assembly, ie.

$$G_w = B_w - (L_w + Sf_w)$$

Where G_w is the glue weight, B_w is the final sample weight, L_w is liner weight and Sf_w is single facer weight.

The size of the samples that were cut out were consistent (to within 0.2 mm). The glue coverage per unit area was calculated using the glue weight (g) and the area (m²) of the corrugated board, ie.

$$G_c = G_w / A_B \quad \text{Equation [5.1.1]}$$

where G_c is the glue coverage in units of g/m², G_w is the weight of glue used and A_B is the area of the board in units of m².

The final step was to measure the average washboarding depth for each sample made using the washboarding profilometer.

G_c , when graphed against washboard depth can be used to determine the relationship between glue coverage and washboard severity for a particular flute.

5.1.2 Washboard depth and glue coverage – Pilot corrugator manufactured

Washboarding depth measurements were made for a range of boards where the pilot corrugator glue gap and speed of manufacture was varied. Papers were conditioned to ISO conditions (see section [5.2.1]) before the washboarding measurements were carried out using the profilometer.

Table [5.1.2] – Basis weight and compositions of papers used in glue quantity investigation - Pilot corrugator manufactured

Paper	Grammage (gsm)	Composition
single facer	205	Kraft Liner Board: mixture virgin softwood (70 %) and virgin hardwood eucalypt (30 %)
fluting	140	Recycled waste – Neutral sulfite semi-chemical (NSSC) pulp
double backer	210	Kraft Liner Board: mixture virgin softwood (70 %) and virgin hardwood eucalypt (30 %)

The materials used included wheat starch (glue), standard C-flute single facer corrugated boards and liners whose composition and basis weight are shown in table [5.1.2].

The glue quantity used in the pilot corrugator manufactured board study was calculated using X-ray fluorescence (XRF) spectroscopy by doping the starch with iron oxide (Fe_2O_3). The concentration of iron oxide used was 0.032 % by weight. This technique has been routinely used at Amcor for many years for determination of applied glue quantity. As the concentration of dopant is low it seems unlikely that the iron oxide will significantly perturb the structure of the starch.

XRF spectrometry involves the detection and measurement of fluorescent X-rays emitted from atoms or molecules (the iron oxide in this case) of a sample that is irradiated by X-rays. The fluorescent X-rays are of a characteristic energy and are detected by an X-ray energy sensitive detector. The intensity of the fluorescent radiation depends on several factors, but is related to the concentration of the element in the sample. More detailed information about XRF spectroscopy can be found in Van Grieken [85]

The number of emitted X-ray photons was measured at the detector and counted. A relationship was then made between the number of photons detected and the concentration of iron oxide present.

A range of glue quantities using the starch and iron oxide was used to arrive at a calibration equation relating the glue quantity for a given count measured by the XRF spectrometer. Glue was applied to the same liner type as was used in the investigation. The paper was weighed prior and after starch application, the difference being the glue quantity applied. The single facer liner with glue together with the corrugated medium and double backer were placed in

the XRF spectrometer to measure an X-ray photon count. This was repeated for samples with no glue up to and including an addition of glue equivalent to approximately 15 g/m. The line of best fit to a plot of the X-ray counts for these samples and the glue quantity was then used as the calibration for the calculation of glue quantity as applied by the corrugator.

The calibration relationship between photon count (x) and glue quantity (y) for the starch/iron oxide mix, liners and medium used was given by the following equation,

$$2.68 \cdot 10^3 \cdot x - 11.0 \quad \text{Equation [5.1.2].}$$

The XRF spectrometer used was manufactured by Refina Instruments, type AB ASFX.

5.1.3 Emulating washboarding using finite element analysis

The geometric model used for the washboarding depth calculation is shown in section [5.4.1(b)]. The geometric model used the C-flute dimensions and is the same model as that used for edge-crush test FEA.

Boundary conditions that simulated the glue are shown in a cross-section of the model in figure [5.1.2]. Applying pressure to elements where glue would normally be located simulated the glue force. The arrows in figure [5.1.2] show the direction and location of the simulated glue force. The percentage glue forces applied were as calculated in section [3.4.2]

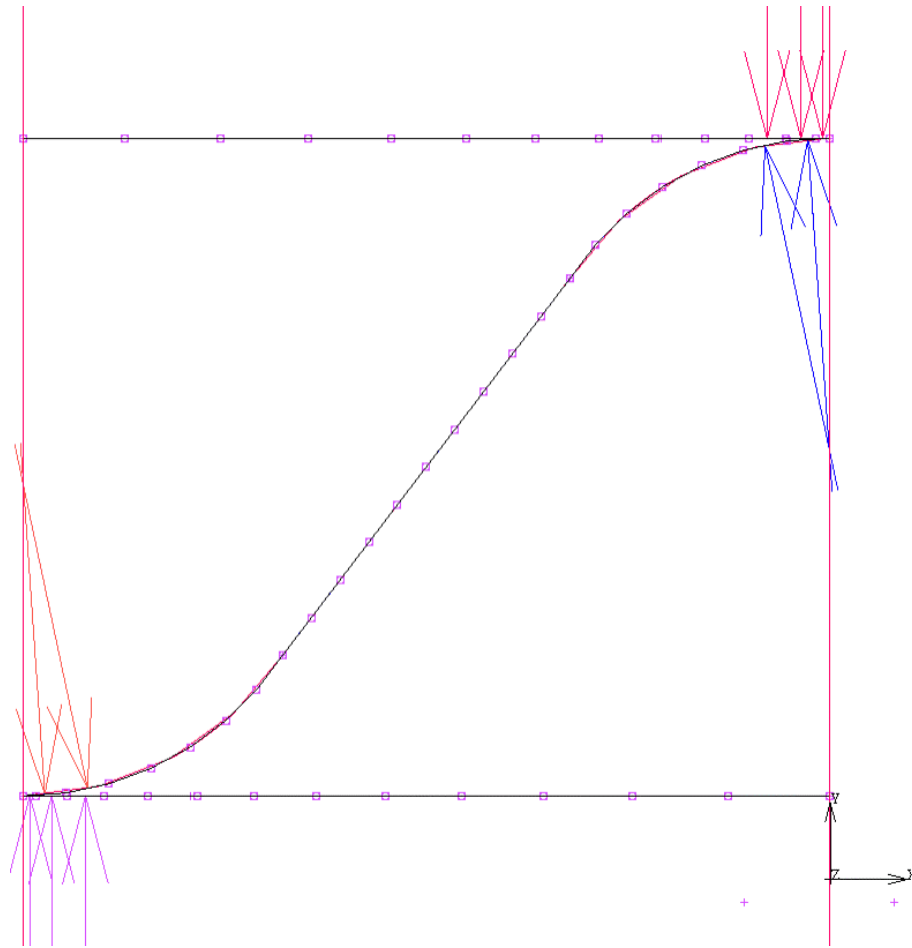


Figure [5.1.2] – Cross section of the washboarding and edge crush model (see figure [5.4.1]) displaying the locations of the simulated glue force by the application of pressure on elements.

Elements that were closer to the tips were smaller to achieve more accuracy as stress concentrates in these regions due to the simulated glue force applied at that location.

Moisture affects the modulus of both paper and glue, and therefore the simulated glue force (see figure [5.1.2]). Paper stiffness and thickness were changed in the various models to simulate this environmental effect (see sections [2.2], [3.2], and [6.3]). The element type used was a four node thick shell with a thickness that was varied from 304 μm to 329 μm . The data used to model paper properties in FEA, including the modulus of paper and thickness, is shown in appendix C for relative humidities from 23 % to 90 %.

The results of these changes were then used to compare empirical results found in the glue coverage and relative humidity investigations

The paper emulated in the FEA was isotropic with a typical Poisson ratio (0.3) of isotropic paper (from Niskanen [54] page 141). The reasoning behind using an isotropic model was purely due to the limited memory and computational power available for the size number of FEA models tested. The fact that isotropic modelling was used and not anisotropic should be kept in mind when comparing empirical results and FEA results presented in Chapter 6.

5.2 Mechanical properties of paper and its effect upon washboarding

5.2.1 Conditioning of paper samples prior to testing.

As discussed in the literature review, paper is sensitive to moisture content, which varies according to temperature and relative humidity. The total moisture content of paper at a specific relative humidity is affected by hysteresis and will depend on whether the moisture content is rising or falling. Therefore samples were conditioned using the following method in an attempt to minimise this.

The papers used were “preconditioned” to the Australian standard AS 1301.414 [30]. This standard prescribes preconditioning to be carried out at a humidity of no more than 35 % RH with temperature largely irrelevant. Preconditioning lasted for a minimum of two days. Testing was performed at ISO lab conditions where relative humidity was held steady at 50 % RH to within ± 2 % RH (three standard deviations) and temperature was held $23\text{ }^{\circ}\text{C} \pm 1\text{ }^{\circ}\text{C}$ (three standard deviations). The samples were held at ISO conditions for a minimum of one week prior to use and testing.

5.2.2 The relationship between the washboarding and the sonic modulus of paper

A study of the relationship between washboarding depth and the sonic modulus of the liner was performed. The sonic modulus was used as an indicator of the elastic modulus. In hindsight, a comparison to Young’s modulus may have been desirable; however, ultrasonic measurement has several advantages over load-elongation testing as discussed in section [2.2.6]

Materials used included wheat starch (glue), standard C-flute single facer corrugated boards and liners of which composition and basis weight are shown in table [5.2.1].

Table [5.2.1] – Basis weight and compositions of papers used in washboarding depth versus elastic modulus investigation.

Paper	Grammage (gsm)	Composition
single facer fluting	280	Top Kraft Liner Board: Top Ply: mixture virgin softwood (70 %) and virgin hardwood eucalypt (30 %)
	150	Bottom Ply: recycled waste Recycled waste
double backer	210	Kraft Liner Board: mixture virgin softwood (70 %) and virgin hardwood eucalypt (30 %)

Apparatus used included an electronic scale accurate to 0.001 g, a hot plate, a glass plate, a steel spreader, 2 kg weight, Nomura Shoji sonic velocity measuring apparatus [86] and the washboard profilometer. The materials were conditioned as described in section [5.2.1]

To achieve a series of liners that comprised the same type and amount of material, samples were cut from the same liner at various orientations to the MD. This gave samples with different moduli and stiffness because the modulus is known to vary as the orientation changes (see section [2.2.6]).

The sonic velocity for a range of orientation angles through the liner was measured using the Nomura Shoji [86]. This instrument measures the sonic velocity by measuring how long sound takes to travel from one of sixteen actuators to sixteen sensors on the perimeter. The sensors are separated by $180^\circ/16 = 11.25^\circ$.

The sonic modulus was then calculated for each orientation using the following equation

$$E_s = \rho \cdot v^2 \quad \text{Equation [5.2.1]}$$

Where E_s is the sonic modulus in Pascal (Pa), ρ is the density of the paper in units kg/m^3 and v is the velocity of sound in the medium in m/s.

The samples used with the single facer board were cut out at intervals of 10 degrees from MD to CD.

The completed and glued samples were then measured using the washboard profilometer to determine the degree of washboard depth. The strain was then calculated from the washboarding depth (see section [3.3]) and the strain calculations were then related to the measured sonic modulus of the paper, and compared with the results obtained using finite element analysis.

5.3 Relative humidity and its effect upon washboarding.

5.3.1 The cycling of relative humidity and its effect upon washboarding

The method described below was developed to investigate how washboard severity and the geometry of the surface profile vary for changing environmental conditions over time. In this section the effect of humidity and temperature cycling upon washboarding in corrugated cardboard samples was studied. In particular, this study showed how the amplitude of washboarding and the shape of washboarding profile changed with environmental conditions and if any of these parameters had a significant hysteresis component.

Equipment used included the profilometer, a corrugated board sample and a programmable environmental room. The C-flute single facer corrugated boards and liners are shown in table [5.3.1] along with the composition and basis weights.

Table [5.3.1] – Basis weight and compositions of papers used in cyclic humidity investigation.

Paper	Grammage (gsm)	Composition
single facer	140	Kraft Liner Board: mixture virgin softwood (70 %) and virgin hardwood eucalypt (30 %)
fluting	115	Recycled waste
double backer	150	Recycled waste

The corrugated board was conditioned to ISO conditions (see section [5.2.1]) before being placed into the environmental room as described below.

The environmental room was programmed to follow the conditions of typical tropical northern Australia and southern Asian environment. The temperature was held constant at 28 °C and humidity was cycled diurnally from a set-point of 50 % RH at approximately 12 noon to 90 % RH at 6 am. The humidity cycling program is shown in figure [5.3.1].

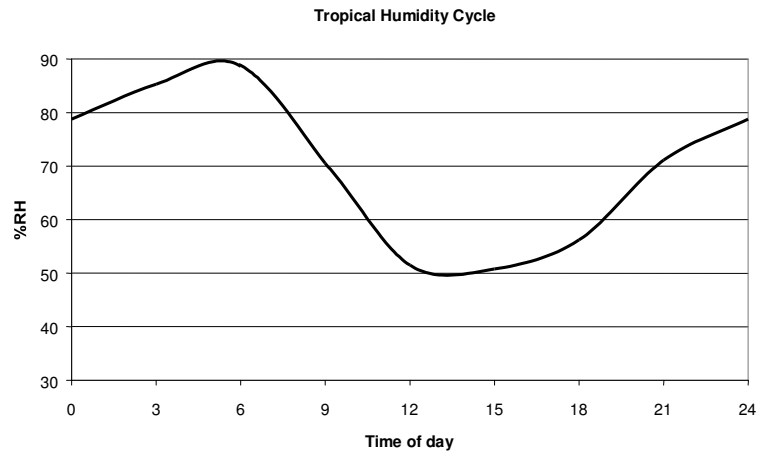


Figure [5.3.1] – Graph of programmed relative humidity cycle of environmental room.

The software used to capture images for the profilometer (see Appendix A) was improved by adding the capability of automatic capture of images at regular intervals. This allowed changes in washboarding depth to be tracked as the environmental conditions changed.

5.3.2 Washboarding depth

The corrugated board sample was first conditioned as described in section [5.2.1] and then placed in the profilometer for measurement of washboarding profile in the programmed environmental room. The room was at 50 ± 2 % RH during its humidity cycling when the sample was placed in the room to minimise possible moisture hysteresis affecting the results. Images of the projected grid lines upon the sample were then taken every thirty minutes.

Once the images had been acquired, the extent of washboarding as the humidity varied was measured. This was achieved by using algorithms as described in Chapter 4 for each image.

5.3.3 Washboarding shape

Discrete Fourier transformation was used to investigate the shape of washboarding (see section [3.1]). The shape of a waveform may be defined by its frequency components. Therefore, a change in relative amplitude of these components relates to a change in shape. A

change in amplitude of a waveform will not change the shape because the relative amplitude of the components does not change (see the multiplication rule in section [3.1.3]). The shape can then be defined as a normalised set of components where each component is divided by the fundamental frequency. The fundamental frequency is also known as the first harmonic and is the second frequency component of a waveform. The first component (also known as the DC component) is the average value of the waveform and has zero frequency.

This means that independent of washboarding depth, the shape can be approximately determined from the normalised frequency components and can be used to reconstruct the fundamental shape of washboarding.

The above methodology was used to study how shape changes with humidity and how it changed in time within an environment of cyclic humidity. The software used for this investigation is described in section [4.2.2(b)]

5.4 The effect of washboarding upon the performance of corrugated cardboard

FEA was used to analyse structural properties and the results generated from FEA were compared to results measured empirically. The correlations of the physical tests with the FEA results allow properties to be tested that can not be done readily using empirical methods. FEA has the added benefit of revealing where the major stresses appear in a board and how changing paper and glue properties affect corrugated board performance. The paper properties and percentage glue forces are the same as described in section [3.4.2] and section [5.1.3]

There are various mechanical tests used to characterise or predict corrugated box performance. The results of three of these were examined in relation to the extent of washboarding depth in this thesis. The three performance measures are investigated both empirically and using finite element analysis (FEA). The performance measures studied were the edge crush test, the three-point bend and MD-Shear.

5.4.1 Edge crush test performance

The edge crush test, also known as edgewise compression testing (ECT), is a test used for characterising corrugated board edge strength in the cross direction (CD). The method used in this thesis conforms to Australian standard 1301.444s-1992 [87] - Edgewise compression resistance of corrugated fibreboard. Samples are compressed and the load that causes failure is noted. It is a measure that correlates with box compressive strength.

5.4.1(a) Empirical Testing

Samples for testing were cut 100 mm long and 25.4 mm wide. Prior to any testing, the samples were conditioned as described in section [5.2.1]. The samples were compressed and the load at failure noted (see section [5.4.3]).

Samples were chosen from a wide variety of corrugated boards with different liners, flute and glue combinations. The degree of washboarding was measured using the profilometer for both sides of each board. Ten ECT test samples were cut from a wide variety of corrugated cardboard boxes and then tested in an ECT compression test apparatus. As this was a random selection of boxes, the exact compositions of the papers remain unknown, though the grammages of the papers were measured as described below.

The various liners and fluting grammages were measured by separation of the three paper components. This was achieved by placing the corrugated boards in a receptacle of very hot water. The papers came apart as the starch lost bonding strength. A microwave oven was used to dry the papers. The moisture content of the paper was then left to equilibrate in ISO room conditions ($25\text{ }^{\circ}\text{C} \pm 1\text{ }^{\circ}\text{C}$ and $50\text{ \% RH} \pm 2\text{ \% RH}$) for one week. The weight was then measured in grams and noted, then divided by the surface area of the board to give units of g/m^2 .

5.4.1(b) Finite element analysis of edge crush test

A C-flute geometric model was defined using MSC.Mentat pre-processor software [88]. This software package enabled elements such as the geometry and boundary conditions to be defined for import into the FEA solver (MSC.Marc [88]). Figure [5.4.1] shows the model with elements defined. The element type used was a four node thick shell with a thickness of $300\text{ }\mu\text{m}$.

Two surfaces of symmetry were defined and shown in figure [5.4.1] as two oblong purple boxes. Also defined was a surface that represents a moving platen, which was used to compress the sample in the z-direction. Force was applied to the model where glue would

normally be (see section [5.1.2]). The nodes at the extreme end of the model, in the positive z-direction, were prevented from moving in the z-direction by boundary conditions, shown in figure [5.4.1].

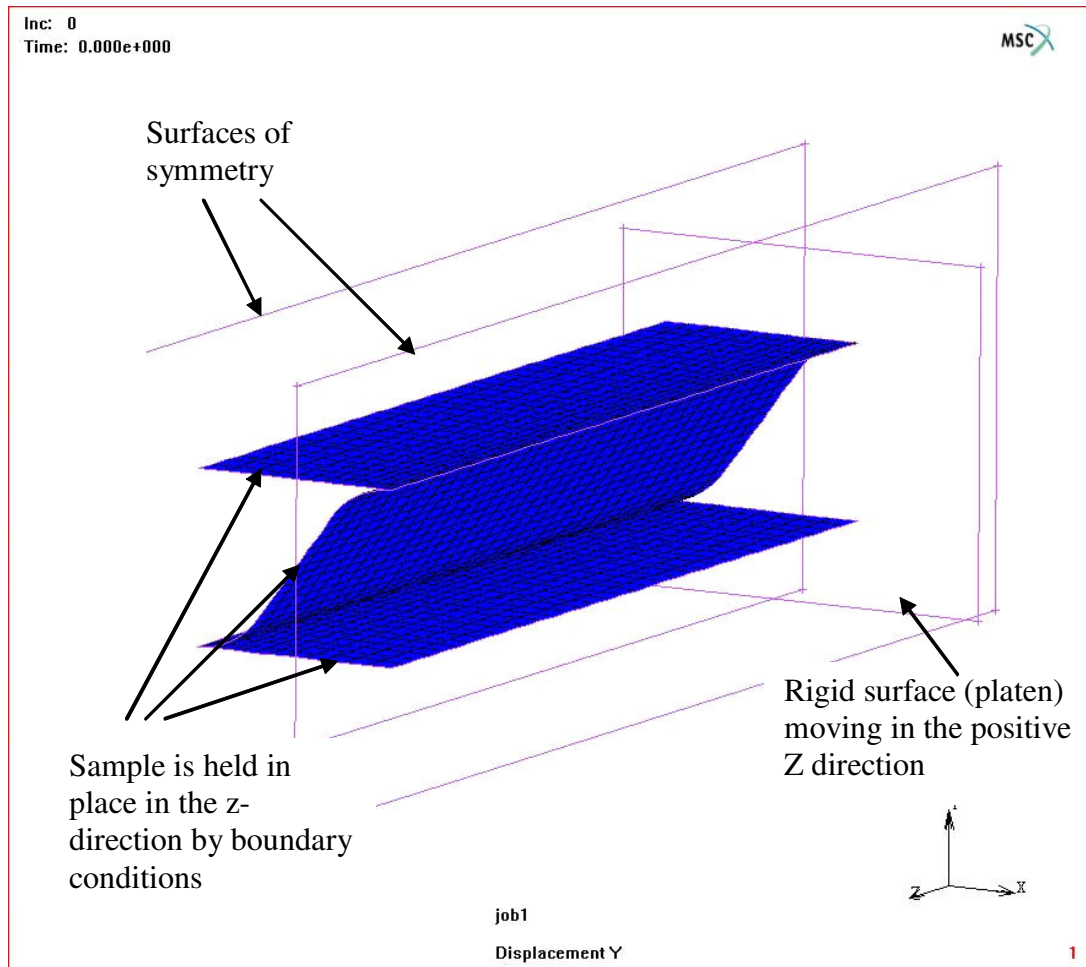


Figure [5.4.1] – Finite element analysis model for washboarding and edge crush test modelling

A change in washboarding depth was investigated by changing the simulated glue force and paper stiffness. As the platen displaces, the load on the sample increases up to a maximum, at which point the load drops again while the platen continues to move. The maximum load at this point is considered to be the load at which the ECT model fails. The results were mapped to see what relative influence these two factors have on ECT results.

5.4.2 MD-Shear performance

MD-Shear is a proprietary test developed by Amcor. In one of the failure modes of a

corrugated box, the medium shears as the box bulges. The MD-Shear test measures the resistance to shearing by twisting a sample through a known angle and the load required for this twist is measured.

The MD-Shear value is calculated using the following equation:

$$M = \frac{3 \cdot \tau \cdot L_{MD}}{\phi \cdot b^3} \quad \text{Equation [5.4.1]}$$

Where M is the MD-Shear value in units of N/m; ϕ is the angle of twist; τ is the torque (in Nm) needed to twist the sample to the angle ϕ ; L_{MD} is the length of the sample; and b is one half of the sample width.

MD-Shear values can be used in box performance prediction. A higher value corresponds to better performance (ie. more resistant to failure by shearing forces).

The test was developed to primarily measure flute damage caused during box conversion of corrugated board, as damage to the flutes significantly reduces box strength [76].

5.4.2(a) Empirical testing

Prior to MD-Shear testing and washboard measurement, the samples were conditioned as described in section [5.2.1]. Samples that were analysed were cut to 25.4 mm widths and 200 mm lengths after washboard depths were measured.

Standard C-flute single facer corrugated boards and liners of which composition and basis weight as used are shown in table [5.4.1].

Table [5.4.1] – Basis weight and compositions of papers used in the MD-Shear and three-point bend investigations

Paper	Grammage (gsm)	Composition
single facer	180	Kraft Liner Board: mixture virgin softwood (70 %) and virgin hardwood eucalypt (30 %)
fluting	115	Recycled waste
double backer	150	Recycled waste

C-flute corrugated samples were made available for these studies that had been manufactured using a corrugator where glue gap settings were changed to vary the quantity of glue applied and hence washboarding produced. The above board was supplied by a previous unrelated study performed by Amcor Research in which starch quantity was varied, the quantity of glue was not measured which did not matter in this study, most important was that washboarding depth varied sufficiently. These samples were also used in the three-point bend study.

The samples used in section [5.1.2] were not suitable for this study as they were manufactured on the pilot corrugator and hence comprised only of single facer and fluting.

Samples were cut from these boards and MD-Shear tests were performed, and the results noted and graphed.

5.4.2(b) Finite element analysis of MD-Shear

A geometric model was developed to investigate the effect of an increase in washboarding on the MD-Shear results. An increase in washboarding was achieved by increasing simulated glue force on the liner and fluting (see section [5.1.2]). The geometric model is shown in figure [5.4.2]

The element type used was a four node thick shell with a thickness of 300 μm . The geometric model was a different size (90 mm long and 25.4 mm wide) than what the normal physical size would be (200 mm long and 25.4 mm wide) due to limitations in computer hardware, as the memory available was not sufficient to handle the quantity of elements appropriate for the normal physical size. One would still expect that a similar result that could be used for relative comparisons, ie. a comparison between an increase in glue in physical testing and an increase in simulated glue force for virtual modelling.

Boundary conditions were applied to the nodes on the right hand side shown in figure [5.4.2] to create antisymmetry on that axis of rotation (twist). This reduced the quantity of elements needed by half and therefore allowed modelling of a larger sample than otherwise possible.

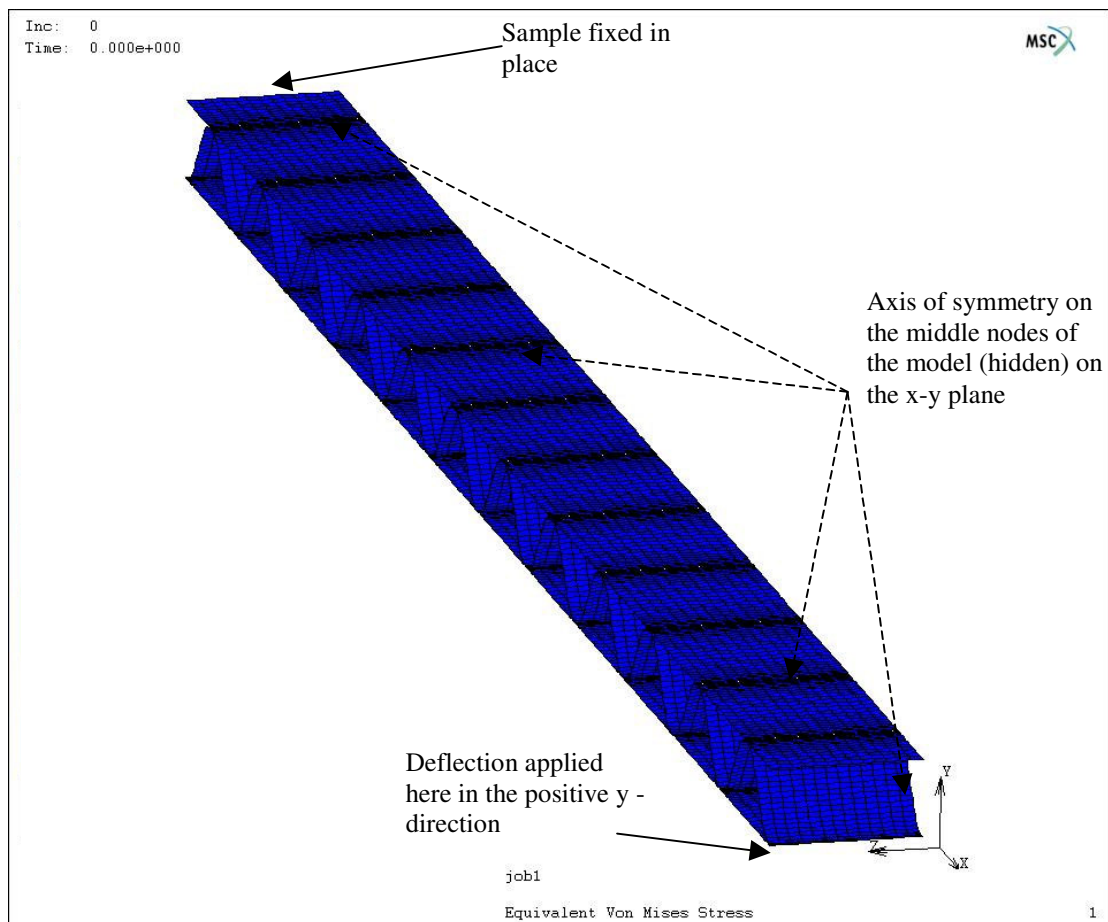


Figure [5.4.2] – Finite element analysis model for modelling MD-Shear.

Elements at the extreme negative x direction of the model were held in place while a point load was applied to the node at the extreme positive x and z direction corner of the geometric model (bottom centre in figure [5.4.2]). At both ends, a half flute length of elements was made very stiff to simulate jaws twisting the sample.

5.4.3 Three-point bend performance

The three-point bend is a classical measurement used to measure bending stiffness of a material or product [38] and is an important measure in characterising corrugated box performance.

The weight of the material contained in a corrugated cardboard box, and any dead load on top of a cardboard box (e.g. such as in a pallet) will cause a box to bulge. As the box bulges, the panels will bend. Three-point bending is a measure that gives an indication of the resistance to this type of failure.

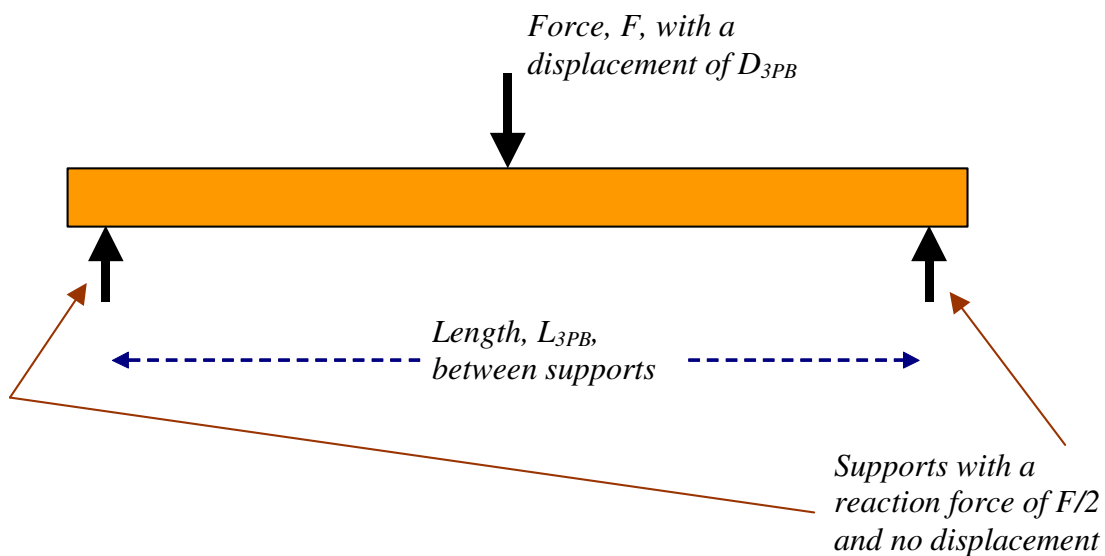


Figure [5.4.3] – A material undergoing a three-point test.

Figure [5.4.3] shows the principle of the three-point test, and equation [5.4.2] shows the relationship between deflection, D_{3PB} , in meters, force, F , in newtons, modulus of elasticity, E , in Pascals and the second moment of Area, I_{3PB} , in m^4 and length, L_{3PB} , in m.

$$D_{3PB} = \frac{F \cdot L_{3PB}^3}{48 \cdot E \cdot I_{3PB}} \quad \text{Equation [5.4.2]}$$

This equation remains valid so long as the material remains in its elastic region. Corrugated boxes, for economic reasons, are generally designed to survive only as long as necessary, with a minimum amount of fibres used, and therefore boxes rarely operate only in the elastic region. Hence, the force needed for the sample to fail and the displacement at this failure point is important, as it gives an indication of corrugated cardboard performance.

5.4.3(a) Empirical testing

The same samples as used in the MD-Shear testing (see section [4.4.2(a)]) were used in an Instron universal testing machine for extracting three-point-bend load versus deflection graphs. Prior to any testing, the samples were conditioned as described in section [5.2.1]. The samples used for three-point bending were 25.4 mm in width and 140 mm in length between centres, and their composition is described in table [5.4.1]. The three-point bend test is described above.

5.4.3(b) Finite element analysis of three-point bend

The geometric model used is shown in figure [5.4.4]. Washboarding depths were varied to study how it affects three-point bend performance. Glue force was simulated at fluting tips, and was varied to analyse a range of models with different washboard depths. The model was 4 mm in width and 100 mm between centres.

A surface of symmetry was defined in y-z plane, seen as a purple box in figure [5.4.4]. Also shown in the figure is a surface defined to act as the middle ‘point’ in the three-point bend test, which moves in the positive y direction over time. Boundary conditions were applied to

the two nodes at the top of furthest fluting tip (top right in figure [5.4.4], preventing movement in the y direction. These two nodes acted as the outside two ‘points’ in a three-point bend test (due to symmetry).

The failure loads were noted for the different washboard depths and compared to results gained from empirical testing.

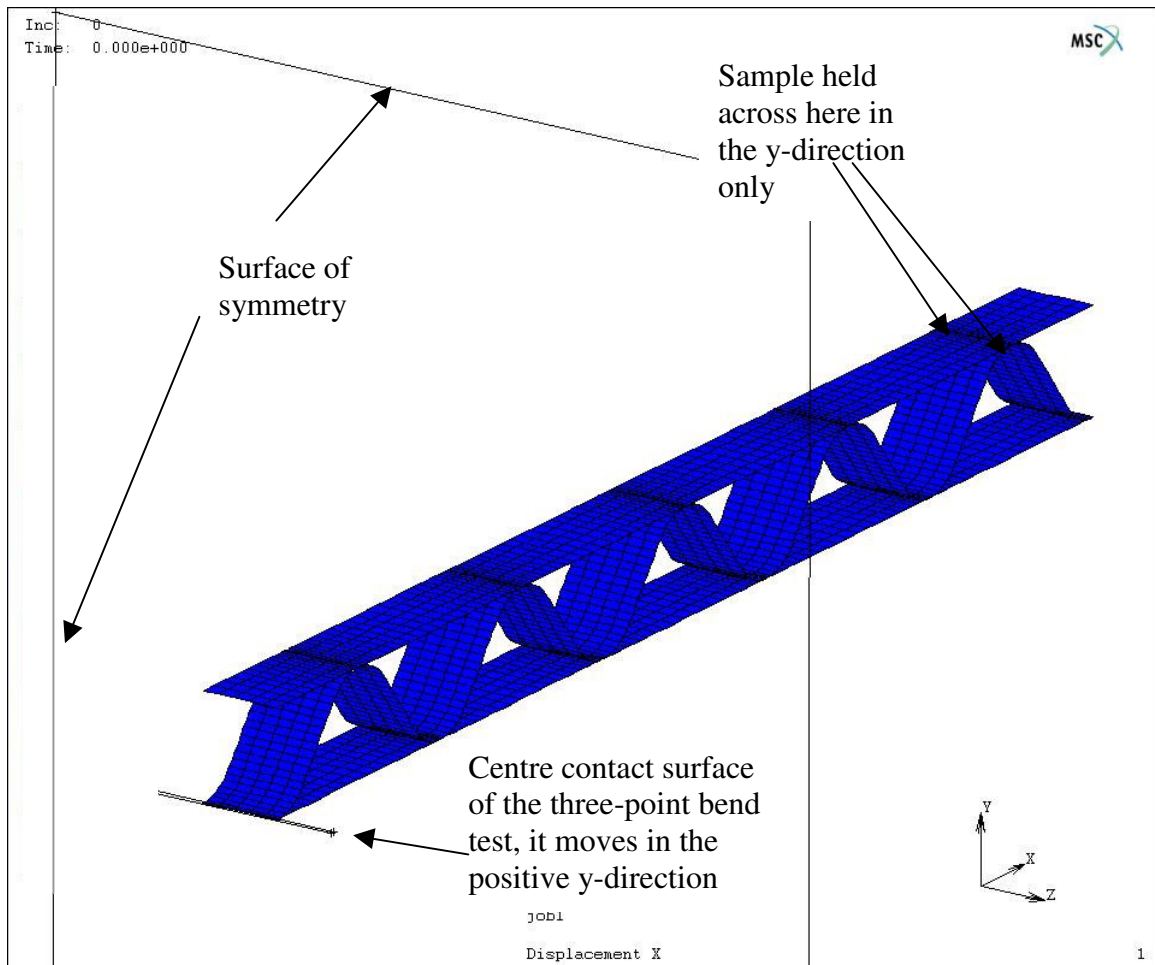


Figure [5.4.4] – Finite element analysis model for modelling three-point bend.

5.5 The effect of washboarding upon the printing quality of corrugated cardboard

5.5.1 Full-tone print

In the past, most methods of measuring print appearance were subjective and were judged by ‘eye’ [3]. To determine how much printing pressure is needed, a reliable reproducible method of measuring the print coverage was developed. The same CCD camera and computer used for washboarding measurement was used to obtain an image to be analysed for print appearance. The software that was developed for measuring print appearance incorporated algorithms (see section [3.5]) that were designed to allow measurements of the print quality that were relatively independent of lighting conditions, colour of the board and ink. The analysis software was included in the main program.

The equipment used for these investigations were the washboard profilometer and associated calibration equipment, a flexographic printer (see section [5.5.2] below), a personal computer incorporating image capturing and analysis software including measurement of print quality and the corrugated box samples (see section [5.5.3] below). The analysis method of measuring print ratio (PR) as used in the software is described in section [3.5].

5.5.2 Flexographic printer settings

Flexographic printing was performed on an RNA-52 printability tester fitted with an optional liquid ink and water feed system accessory and anilox roller. The RNA- 52 was manufactured by Research North America Inc.

The following flexographic settings were used to print the samples:

- Average printing force: 350 N;
- Anilox roller screen : 120 lines/cm;

- Anilox volume 5.5 ml/m²
- Average anilox force: 50 N;
- Printing strip size: 50 mm wide by 200 long;
- Printing Speed: 0.6 m/s

The ink had the following properties:

- Water based, standard red flexographic ink supplied by Research North America Inc.;
- Viscosity: 20 ± 0.5 seconds using a Zahn cup #2 at a temperature of 23 °C;
- Quantity: Defined by the Anilox roller (120 lines/cm)

Samples were printed at ISO conditions (50 ± 2 % RH and 23 ± 1 °C)

5.5.3 Sample selection

Various samples were collated for the study of the print quality of full-tone print onto washboard affected board. The furnish of the samples tested varied from recycled to virgin and the surface treatment of the samples varied from non-bleached to bleached board.

The samples used are shown in table [5.5.1]. These samples were cut to dimensions of 200 mm by 140 mm and were subsequently measured for washboard severity using the profilometry method described in Chapter 4 and conditioned as described in section [5.2.1]. The samples were then cut in half to form 2 samples of size 200 mm by 70 mm. This is the size suitable for use in the flexographic printer. This allowed two independent measurements to be made of the washboarding and print quality for each board.

Table [5.5.1] – Washboarding samples used for the full-tone printability study.

Sample Number	Outside Liner/Flute(Type)/Inside Liner*	Description
1	WL151/FS136C/TK150	Bleached
2	WL151/FS136C/TK150	Bleached
3	TK280/TL180C/TK280	Unbleached
4	BW220/PG160C/TK243	Bleached- CHH**
5	TK280/TL180C/TK280	Unbleached
6	TK120/FL115B/TK120	Unbleached, 20% waste
7	TK120/FL115B/TK120	Unbleached, 35% waste
8	TK120/FL112B/TK120	Unbleached, Uncalendered
9	WL200/FS160C/KL204	Unbleached
10	TK150/FS160C/KL170	Unbleached
11	WL200/FS160C/KL210	Bleached
12	WL200/FS160C/KL250	Bleached
13	WL194/FS160C/TK243	Bleached, 2 ply trial

*The Amcor nomenclature in table [5.5.1] are as defined below:

- TK – Top Kraft
 - Top ply: mixture virgin softwood (70 %) and semi-chemical hardwood eucalypt (30 %);
 - Bottom ply: recycled waste.
- WL – White Liner - Virgin pulp hardwood (Eucalypt).
- FS – Strong fluting – Recycled waste NSSC pulp (neutral sulfite semi-chemical).
- FL, TL – Recycled waste.
- KL – Kraft Liner - mixture virgin softwood (70 %) and virgin hardwood (30 %).

C and B after the flute name refers to the flute type (see section [1.1.1]). The numeric values are the nominal grammages.

** CHH – BW220/PG160/TK243 is Carter Holt Harvey nomenclature and is not known to the author to what they represent.

Chapter 6 - Results and Discussion

6.1 Manufacturing process of corrugated cardboard and its relationship to washboarding geometry

6.1.1 Washboard depth and glue amount

6.1.1(a) Manual glue application

Table [6.1.1] shows the results for measurements of the washboarding depth for a given quantity of glue per square meter applied, and the washboarding data are represented graphically in figure [6.1.1]. The corrugated cardboard consisted of a single facer and manually attached double backer. The washboarding depth was measured on the double backer side. The paper basis weight and composition of the papers used are shown in table [5.1.1].

Table [6.1.1] - Results of manual glue study.

Sample No	Total Weight (no glue) g	Final Weight g	Glue Weight g	Glue quantity g/m ²	Washboarding um
1	16.78	17.15	0.37	12.33	112
2	16.95	17.54	0.59	19.67	166
3	16.79	17.53	0.74	24.67	214
4	16.70	17.51	0.81	27.00	200
5	16.92	17.35	0.43	14.33	140
6	16.97	17.25	0.28	9.33	106
7	16.83	17.39	0.56	18.67	156
8	16.95	17.57	0.62	20.67	203
9	17.62	18.29	0.67	22.33	216
10	17.04	17.53	0.49	16.33	139
11	17.01	17.61	0.60	20.00	150
12	16.85	17.68	0.83	27.67	213
13	17.00	17.86	0.86	28.67	233

The results displayed an approximately linear relationship between washboard depth (y) and glue quantity (x), where the line of best fit was given by the equation $y = 6.59x + 40.2$ with a correlation value, R^2 of 0.87.

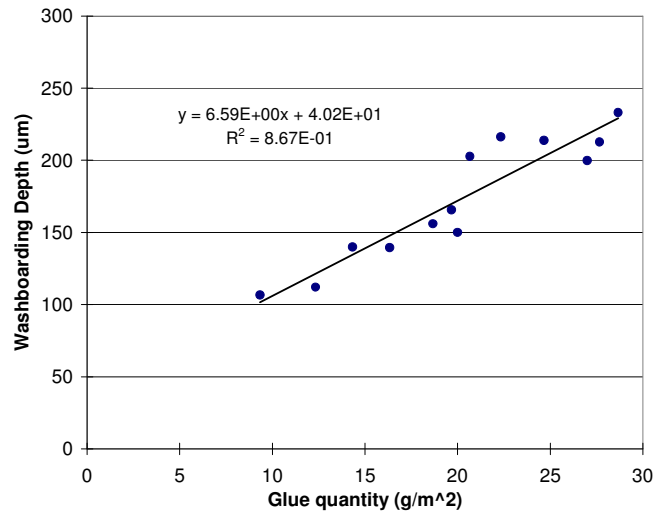


Figure [6.1.1] –Relationship between glue quantity and washboarding depth for manually glued liners.

6.1.1(b) Corrugator glue application

A trial where the glue (starch) quantity was varied by changing the glue gap on the pilot corrugator was performed. The glue was doped with iron oxide and X-ray spectroscopy was used to determine the starch quantity as described in section [5.1.2].

Table [6.1.2] - Results of the corrugator glue study.

Glue Gap (units)	Manufacturing Rate (m/min)	Glue (g/m ²)	Washboarding (um)	Standard Deviation (um)
100	60	5.43	52	3
100	100	6.36	65	5
100	140	8.04	72	4
100	180	8.99	74	4
205	60	6.07	62	3
205	100	7.03	75	9
205	140	8.71	78	3
205	180	9.38	93	6
303	60	6.67	64	3
303	100	7.61	74	4
303	140	8.68	85	6
303	180	10.05	110	7
419	60	7.71	77	5
419	100	8.37	79	4
419	140	9.11	81	1
419	180	10.31	105	1

In addition the machine speed was also varied to investigate how these settings affect washboarding severity. The paper basis weight and composition of the papers used are shown in table [5.1.2]. The results of this study are shown in table [6.1.2] and figure [6.1.2].

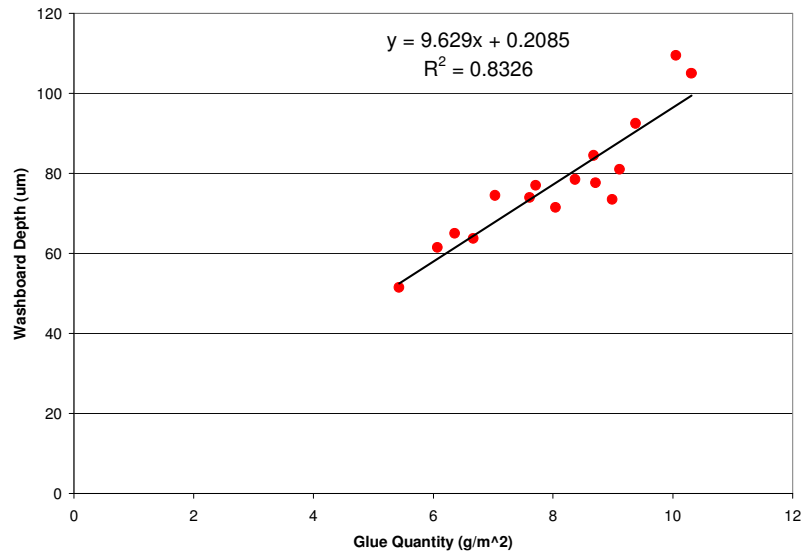


Figure [6.1.2] - Relationship between glue quantity and washboard depth for corrugator manufactured board.

These results show that for machine manufactured board, an increase in glue quantity results in an increase in washboard severity (as was the case with manually manufactured board in the previous section). In this case, the relationship shown in figure [6.1.2] between washboarding depth (y) for a given glue quantity (x) was also approximately linear and was given by the equation $y = 9.63x + 0.209$ with a R^2 value of 0.83.

If both the machine manufactured board and manually glue board results are plotted on a single graph (figure [6.1.3]) we see that a linear relationship between glue quantity and washboard depth can also fit this extended set of data

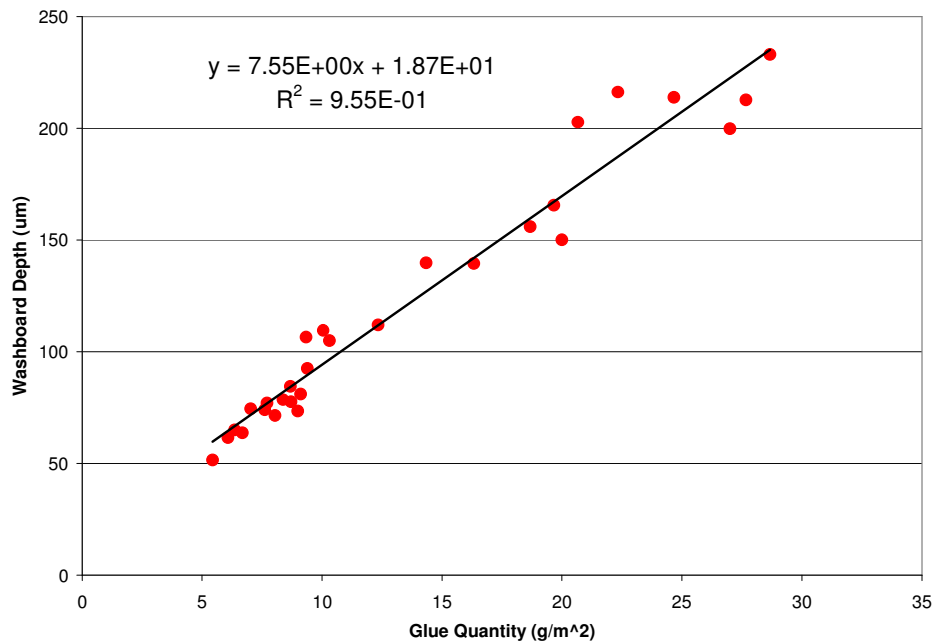


Figure [6.1.3] - Relationship between glue quantity and washboard depth for corrugator manufactured board and manually glued liners.

This data suggests that, in the case of washboarding depth, the manually constructed corrugated cardboard (fluting furnish- Recycled 150 g/m²) behaves in a similar fashion to machine manufactured board (fluting furnish- NSSC pulp 140 g/m²) as the glue quantity is varied, despite the differences in fluting furnish for the two sets of samples.

6.1.1(c) Machine speed

The measurements presented in Table [6.1.2] indicated that as the machine speed increased the quantity of glue applied also increased. Figure [6.1.4] illustrates the results for various speeds and glue gap settings.

The mechanism for the increase in glue quantity accompanying an increase in speed was not investigated as part of this thesis but may be partly explained by an increase in shear and hydrodynamic effects due to the viscous behaviour of the glue as the speed increased, altering the roll pick-up of glue. On recently designed corrugators, the glue gap which sets the amount

of glue is automatically changed as the speed increases or decreases in an attempt to control the adhesive pick up.

The results, however, from table [6.1.2] and figure [6.1.4], show for all the machine speeds investigated, as the glue amount was increased, washboarding increased linearly with it.

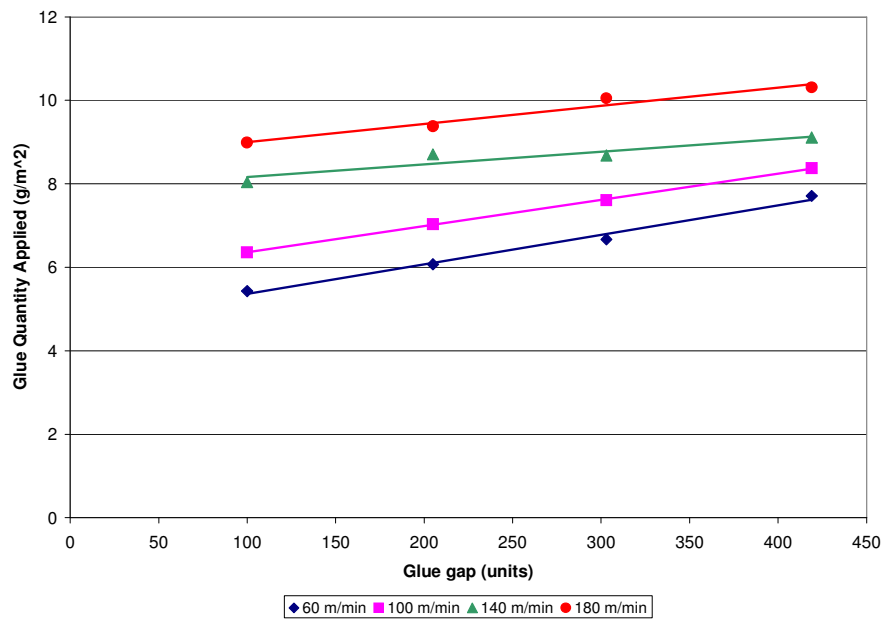


Figure [6.1.4] – Relationships between corrugator glue gap, speed and applied glue quantity.

6.2 Mechanical properties of paper and its effect upon washboarding

6.2.1 Washboarding and its relationship to the elastic modulus (measured ultrasonically) of paper

The effect of changing the orientation of a liner glued manually to single facer board with respect to the machine direction (MD) was studied using techniques described in section [5.2.2]. The quantity of glue used in this study was on average $3.8 \pm 0.1 \text{ g/m}^2$ and the paper basis weight and composition of the papers used are shown in table [5.2.1].

Starch was applied to the single facers by using the same film of glue to minimise any effects upon the washboarding depth due to changes in glue quantity. Table [6.2.1] shows the results for this study. Also shown is the speed of sound through the paper, the elastic modulus (measured ultrasonically), washboard depth and average strain for the given orientations of the liner. The elastic modulus was calculated by measuring the speed of sound through the paper and the strain was calculated from equation [3.3.1] in section [3.3] using the washboarding depth.

Table [6.2.1] – Results of the liner orientation study

Orientation (deg)	Sound Velocity (m/s)	Sonic Modulus (Gpa)	Strain (%)	Washboarding (μm)
MD	3411	9.3	0.00600	33
10	3276	8.6	0.00662	35
20	3134	7.9	0.00682	36
30	2960	7.0	0.00820	42
40	2760	6.1	0.00914	43
50	2589	5.4	0.01079	47
60	2453	4.8	0.02106	69
70	2348	4.4	0.03015	85
80	2313	4.3	0.03317	89
CD	2296	4.2	0.04712	108

The washboarding depth was found to increase in a non-linear fashion for the same approximate glue quantity as the sonic modulus decreased (see figure [6.2.1]).

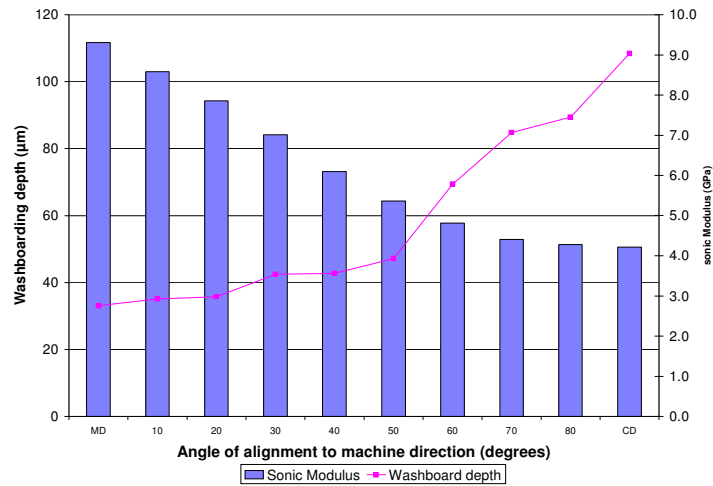


Figure [6.2.1] – Increasing washboarding depth as the sonic modulus (measured ultrasonically) decreases.

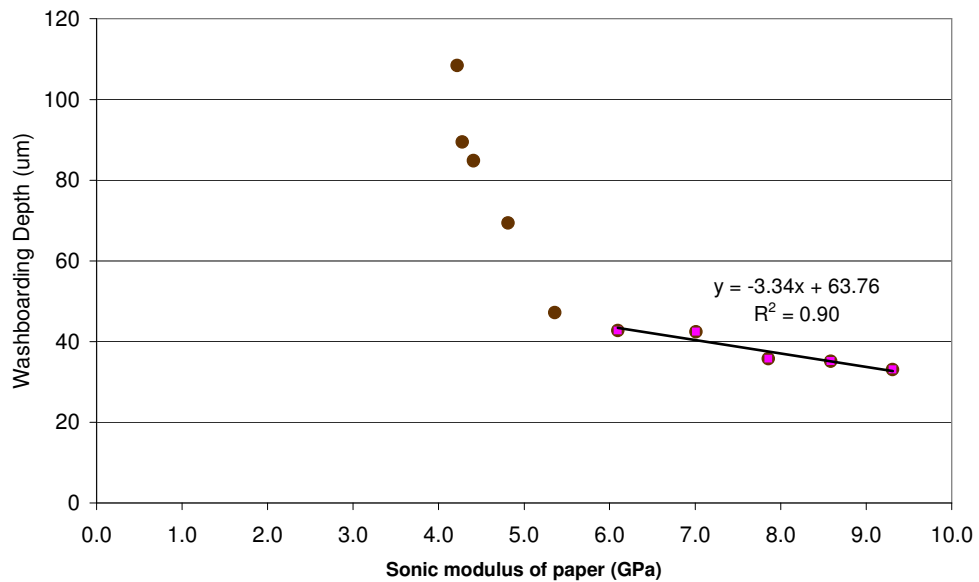


Figure [6.2.2] – Relationship between sonic modulus (measured ultrasonically) of the paper and washboarding depth.

A pronounced non-linear relationship for washboarding depth is seen for sonic moduli values less than 5 GPa in figure [6.2.2]. The non-linearity may be due to the paper entering a plastic deformation region as the paper is orientated further away from the MD, for the quantity of glue applied (3.8 g/m^2). It will be shown later, in section [6.2.4(a)] that finite element modelling of washboarding is consistent with this sharp rise in washboarding.

A typical range of the elastic modulus, measured ultrasonically, of packaging grade paper is in the range from 4 to 10 GPa. For a paper with low stiffness, corresponding to a elastic

moduli (measured ultrasonically) of less than 5 GPa, the washboarding depth increases rapidly. It is therefore advisable that the paper stiffness should be sufficiently high (> 5 GPa) for the paper to remain in its linear region, and the severity of the washboarding may become excessive for relatively small increases in glue quantity. It is unclear how much of this rapid increase in washboarding depth is due to the fibre alignment and/or the built in stresses of the paper. However the relationship between stiffness (regardless of why it is different) and force imparted by the starch should theoretically remain true. Further study is needed to confirm this hypothesis.

6.2.2 Washboarding and its relationship to strain

Figure [6.2.5] shows a typical washboarding profile for a C-flute corrugated board obtained from a single washboarding profile. The amount of strain for a given washboard depth can be approximated from such profiles using the piecewise approximation as described in section [3.3] only if the distance between peaks (or washboarding frequency) remains the same irrespective of the degree of washboarding.

To test whether the assumption of constant washboarding frequency is valid, the washboarding profiles for the same sample which had differing amounts of washboarding (as a result of changes in humidity - see section [6.3.1]) were examined using Fourier analysis.

Figure [6.2.3] shows graphs of the frequency components for four samples for which the washboarding changed over a range of approximately 60 %. The peak positions are almost identical and after scaling the frequency components (see figure [6.2.4]) the shapes of the peaks are also almost identical. This strongly suggests that the assumption of constancy of the distance between washboarding peaks is valid for this range of washboarding.

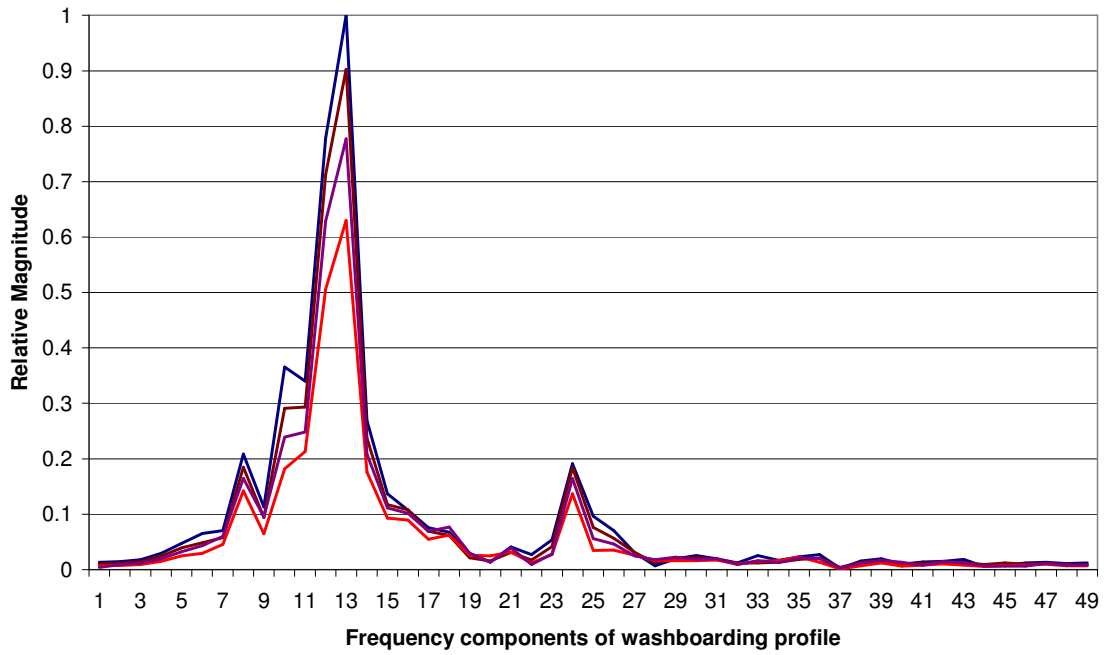


Figure [6.2.3] – Relative frequency components calculated using Fourier transformations of a range of washboarding profiles with different depths, where the depth for the blue curve is 60 % greater than for the red curve.

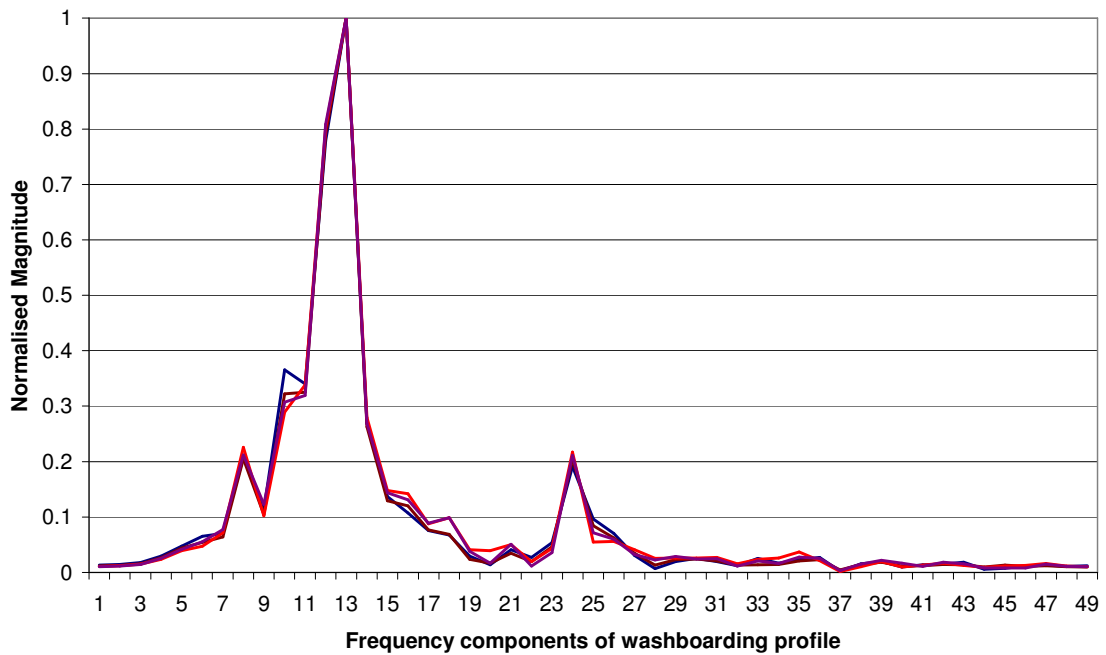


Figure [6.2.4] – Normalised Fourier transformations of a range of washboarding profiles with different depths, where the depth for the blue curve is 60 % greater than for the red curve.

Strain was calculated for a large selection of washboarding profiles that had been collected for other studies by using equation [3.3.1], including boards used in the starch quantity investigation.

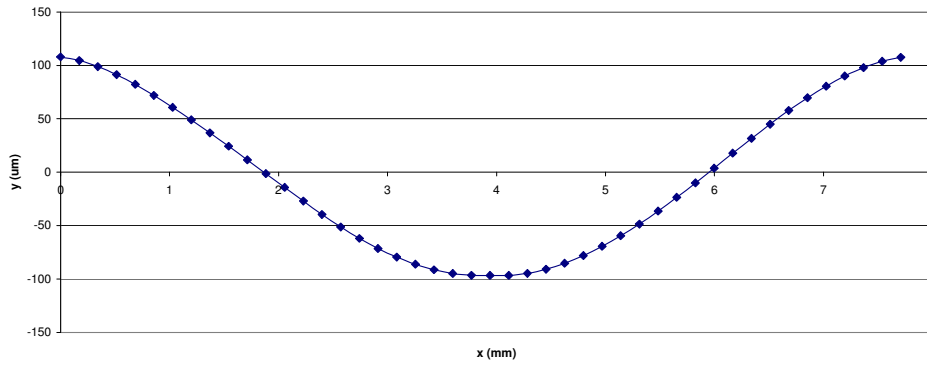


Figure [6.2.5] – A typical washboarding profile used for strain calculation

The relationship, for the case of C-flute, of the strain for a given washboarding depth was found to best fit a power law given by,

$$\varepsilon = 8.54 \cdot 10^{-6} \cdot W_d^{1.85} \quad \text{Equation [6.2.1],}$$

where ε is the strain expressed as a percentage and W_d is the washboard depth in μm . The correlation is high with a R^2 value of > 0.99 .

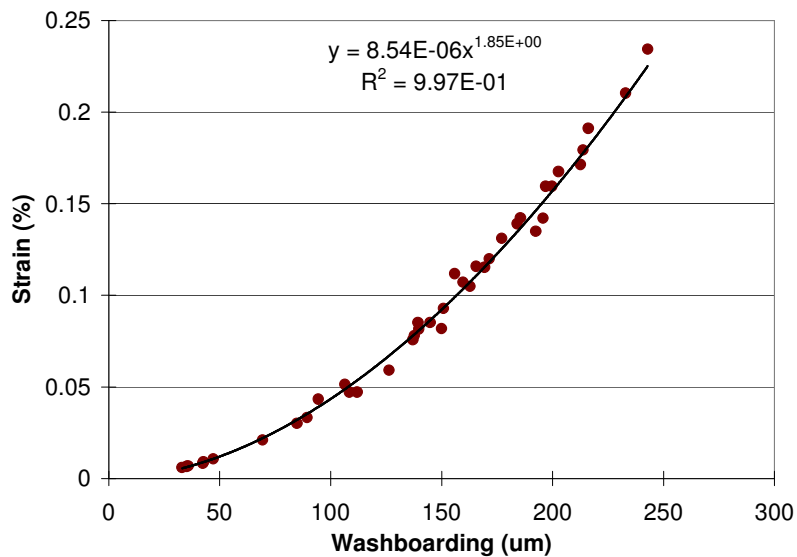


Figure [6.2.6] – Relationship between washboard depth and liner strain.

The empirical relationship between washboarding and strain is illustrated in figure [6.2.6]. Using equation [6.2.1], the strain was calculated for the given starch quantities and is illustrated in figure [6.2.7]. The relationship between glue quantity (x) and strain (y), given by

a least squares power law fit is $y = 9.21 \cdot 10^{-4} \cdot x^{1.62}$ with a correlation, R^2 value of greater than 0.96.

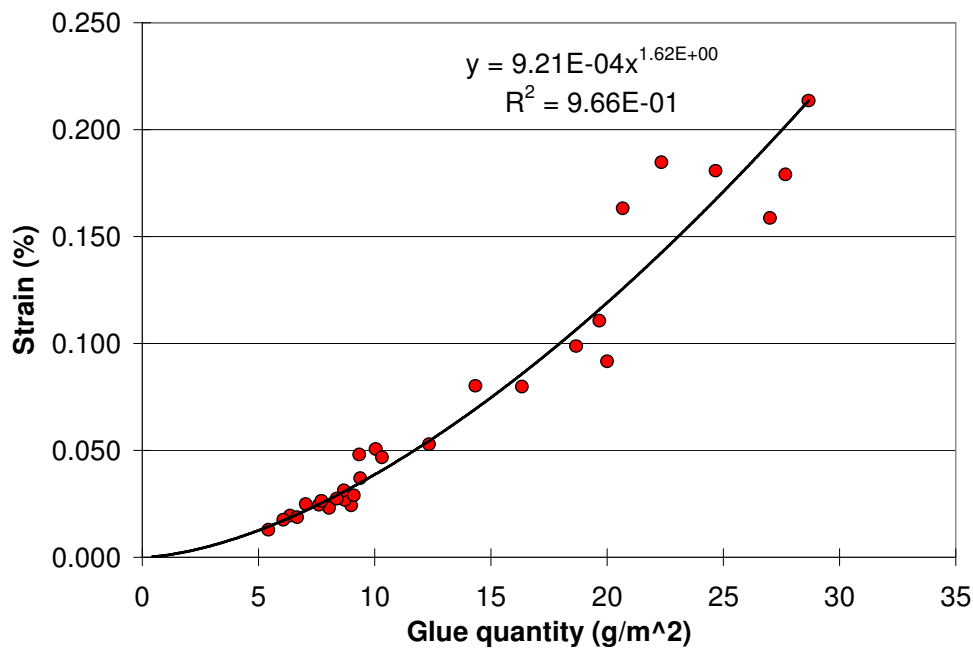


Figure [6.2.7] – Relationship between glue quantity and washboarding liner strain.

6.2.3 Finite element analysis of washboarding

A finite element model displaying washboarding after the application of simulated glue force is shown in figure [6.2.8], where the colour map illustrates the change in displacement of the elements and nodes from the undeformed model, while figure [6.2.9] shows the concentration of stresses due to the boundary conditions simulating glue application forces. Figure [6.2.10] shows a side on view to more clearly illustrate the washboard deflection. Figure [6.2.11] is taken from the same view except deflections are exaggerated by a factor of ten.

It can be seen from figure [6.2.11] that the thickness of the corrugated board (ie. the distance between top and bottom liner) decreases as both the fluting and the liner deform. This would indicate a reduction in bending stiffness, as stiffness is proportional to the cube of thickness of the material (see section [3.4.1(e)]). It is also noticeable that the liner deflects considerably more from its original shape than the fluting. The blue colour of the top liner indicates a

displacement in the negative y direction while the bottom liner is yellow, indicating a displacement in the positive y direction. The red colour of the fluting indicates minimal displacement.

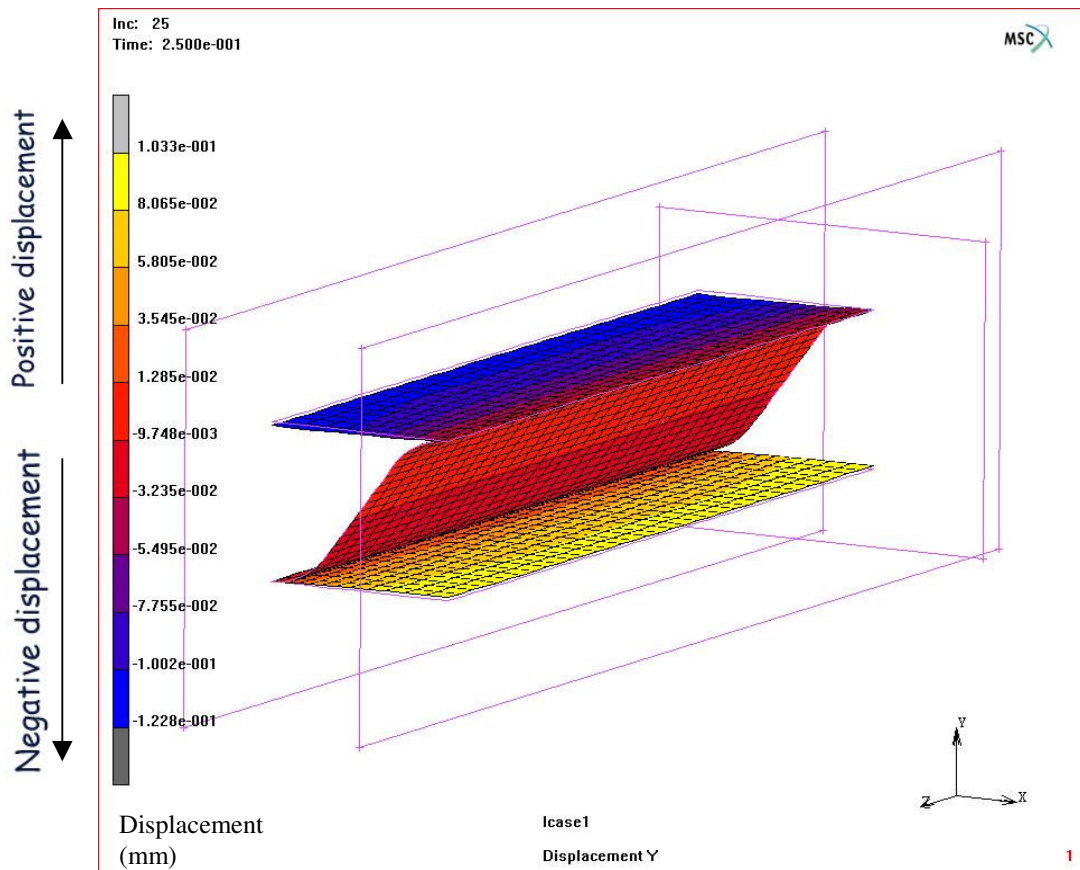


Figure [6.2.8] – Finite element model showing washboarding due to simulated glue force application.

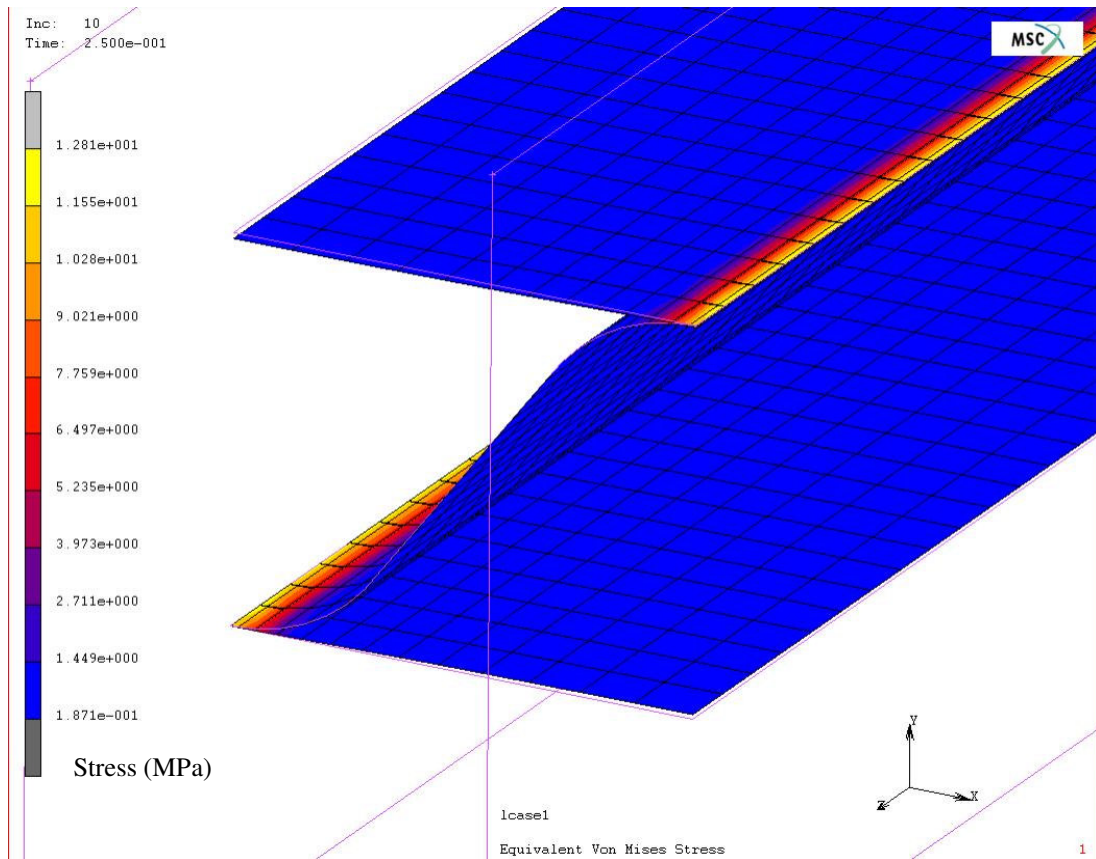


Figure [6.2.9] - Finite element model showing stress concentration and washboarding due to simulated glue force application.

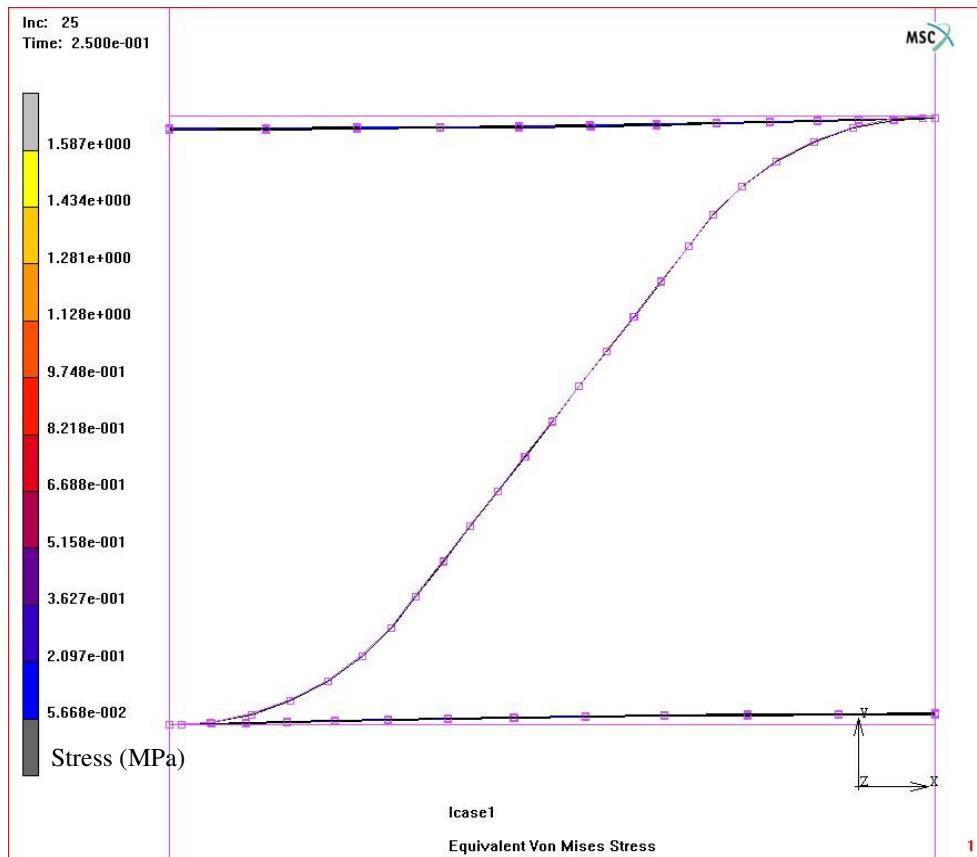


Figure [6.2.10] - A side on view of the finite element model depicting washboarding.

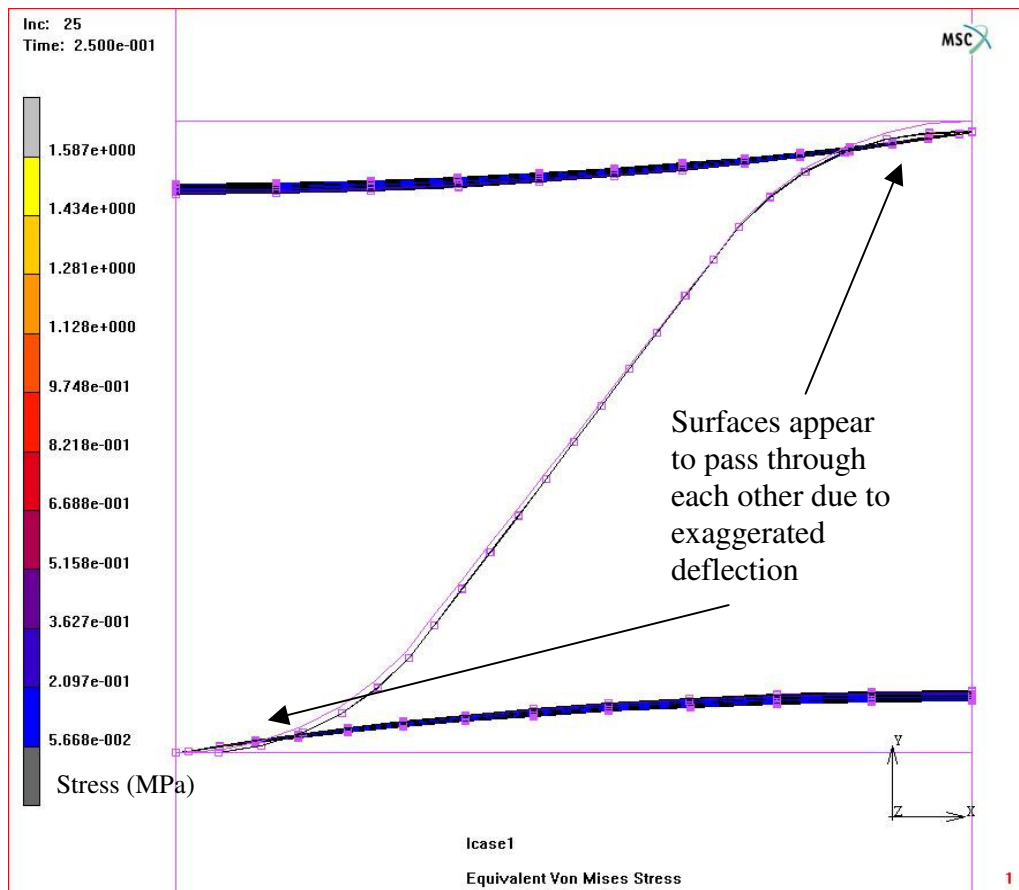


Figure [6.2.11] – An exaggerated view of washboarding and fluting deformation (10x normal displacement) illustrating a reduction in corrugated cardboard thickness.

5.2.4(a) Elastic modulus of paper

In this section a finite element study is presented examining how washboarding depth varies for given paper properties and glue applied. The simulated glue forces, applied by boundary conditions, were chosen such that they would produce a typical amount of washboarding, in this case approximately 50 μm for an elastic modulus of the paper of 6450 MPa. The paper properties used in the modelling were varied to study how washboarding depth would be affected by changing the modulus of the simulated paper while keeping the thickness constant at 310 μm . The modulus of elasticity for the simulated papers were chosen to be 7060, 6450, 5800 and 5190 MPa.

A dramatic increase in washboard depth was observed in figure [6.2.12] for an Young's modulus less than 5500 MPa. The dramatic increase was observed empirically earlier in the

study of liner orientation with respect to machine direction (see figure [6.2.2] in section [6.2.1]), where a similar increase was observed for orientations of the paper that corresponded to a sonic modulus of less than 5000 MPa.

It is important to note that a direct comparison between the elastic modulus (measured ultrasonically) and the simulated elastic (Young's) modulus is not possible without considering the differences in the measurement of sonic modulus and mechanical elastic modulus (ie. Young's modulus). In this case if we assume that the measured sonic modulus was 20 % greater than Young's modulus (as discussed in literature review section [2.2.6]) then the sharp rise seen in figure [6.2.12] was closer to an equivalent sonic modulus of 6500 MPa compared to the 5000 MPa value obtained from the physical testing. Further FEA modelling to obtain more washboarding depth values for elastic moduli in the range of 5000 to 6500 MPa could be performed to verify the discrepancy.

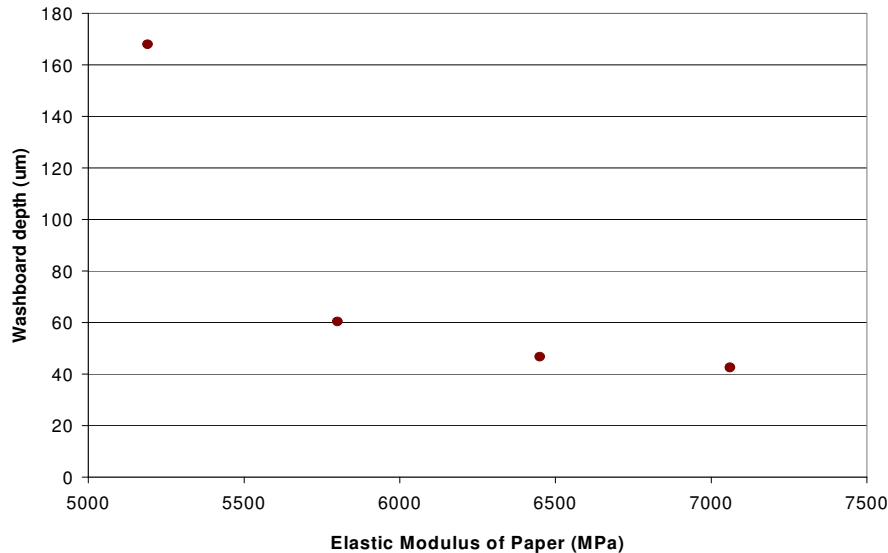


Figure [6.2.12] – Relationship between paper modulus and washboarding depth using finite element analysis.

From this information it is clear to see that measurement of the sonic modulus is insufficient to predict the amount of washboarding that will be produced for a particular paper and applied glue, unless the glue applied is minimal (which is the clear aim for commercial manufacturers). When the elastic modulus is low the force exerted by the glue may be large

enough for the paper to enter the non-linear region of the stress-strain curve and further information about the stress and strain characteristics, other than the sonic modulus of paper, may therefore be needed to predict washboarding depth.

5.2.4(b) Simulated glue force

Finite element analysis was used to study how changes in simulated glue force affect washboarding, while keeping the paper properties constant. Figure [6.2.13] shows the result for a typical paper at 50 % RH, a thickness of 310 μm and an elastic modulus of 6450 MPa. The boundary conditions used to simulate the glue force (see section [3.2]) between liner and fluting was increased from zero to a force needed to create a washboarding depth of up to approximately 150 μm , which corresponds to the defined simulated glue force of 100 %, as seen in figure [6.2.13]

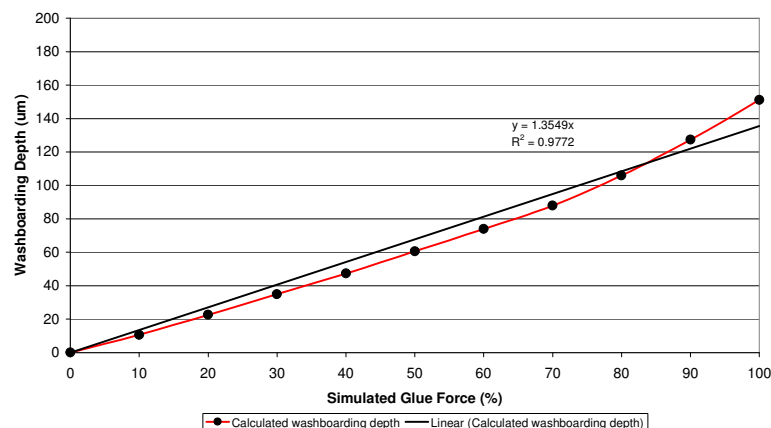


Figure [6.2.13] – Relationship between glue pressure and washboarding depth using finite element analysis.

In this figure we observe a non-linear response which also can be approximated to a linear response with a mean deviation of 10 % and a maximum deviation of 15 μm . In comparison, the maximum deviation from linearity observed was 29 μm in the experimental investigation of glue quantity versus washboarding depth (section [6.1.1]) for the same range of washboarding depths. This maximum deviation seen above fits well within the maximum deviation seen experimentally, therefore little can be concluded about the non-linearity of the data in figure [6.2.13]

Repeating the corrugated manufactured study with a greater range of glue quantities may reveal more information in regards to the result of the FEA investigation above. However since washboarding was rarely seen to be above 120 μm for commercial board the non-linearity seen in figure [6.2.13] may not be important.

6.3 Environmental conditions and its effect upon washboarding geometry

6.3.1 The cycling of relative humidity and its effect upon washboarding

6.3.1(a) Washboarding depth and creep

The following study was performed before a reliable calibration method (see section [4.2.5]) was developed by the author and therefore absolute washboard measurements are not presented. The relative changes in washboarding still apply. The paper basis weight and composition of the papers used are shown in table [5.3.1].

Figure [6.3.1] shows the result of measuring washboarding for a corrugated sample placed in an environment of cyclic humidity from 50 % to 90 % RH (see method section [4.1] for details). It is evident that the washboarding depth reduces as relative humidity increases. Also evident, from the gradual drift upwards, is creep in the liner or starch as the number of cycles increase over time.

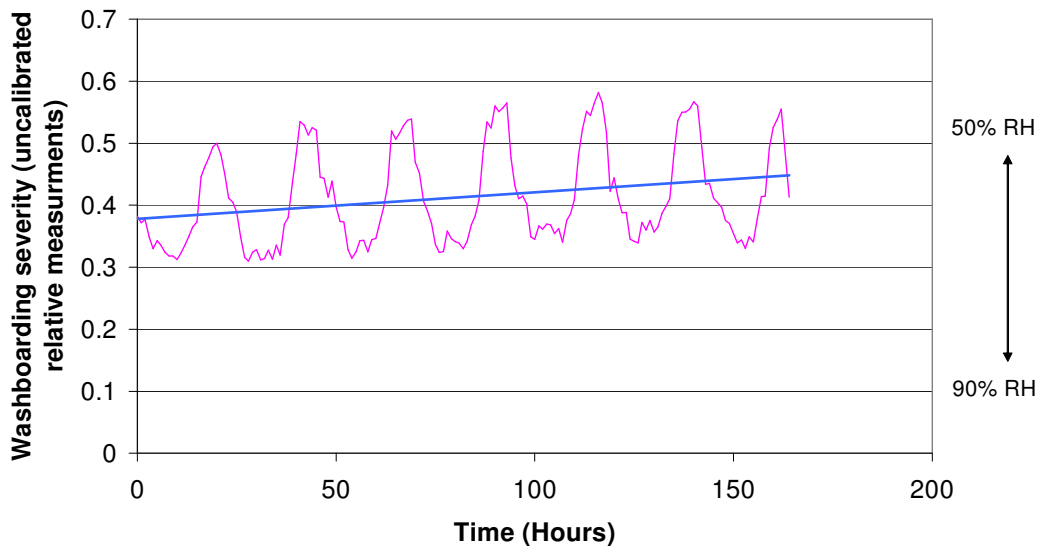


Figure [6.3.1] – Change in washboarding depth over time due to the cycling of relative humidity. Also shown is the trend line (blue) indicating the drift upwards attributable to creep.

One may assume that, because paper stiffness decreases with increasing moisture, the severity of washboarding should increase. However, there are two other factors that may influence

this result; hygroexpansion of the board, and the elastic modulus of the glue. The bending stiffness of paper is increased as paper thickness increases due to hygroexpansion (see section [2.2.3]) while glue stiffness decreases with increasing relative humidity (see section [3.2.2]). These factors together actually reduce the severity of washboarding. Modelling using finite element analysis confirms this result and is used to investigate this further, as presented in section [6.3.2].

6.3.1(b) Washboarding shape and moisture content

In this section the results of an investigation of how moisture content affects the washboarding profile is presented. The method of extracting an average shape was performed as described in section [5.3.3].

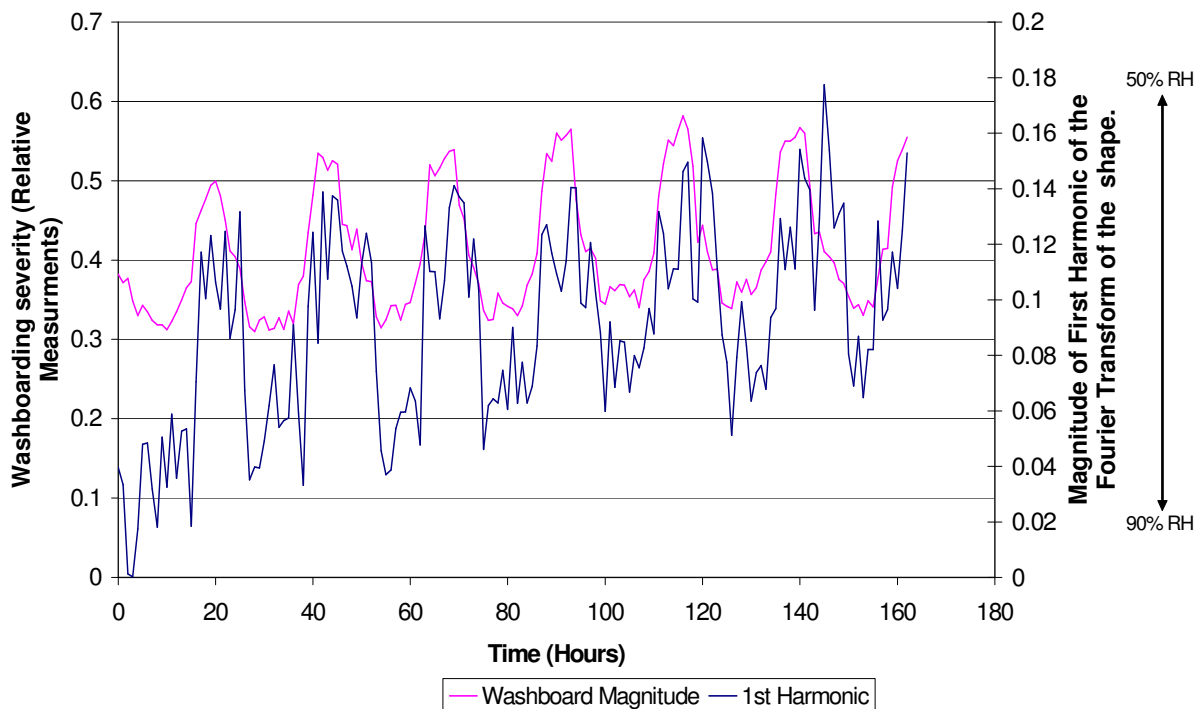


Figure [6.3.2] – The relationships between the magnitude of the first harmonic, cyclic relative humidity and washboarding depth.

The relationship between the magnitude of the first harmonic in relation to the changing relative humidity and washboarding depth is shown in figure [6.3.2], (see section [5.3.3] for

the significance of the first harmonic). The following two graphs show the shape at specific times during the cycling of relative humidity. The first, figure [6.3.3], shows the shape of washboarding for the relative humidities of 50, 65 and 90 % for a single board.

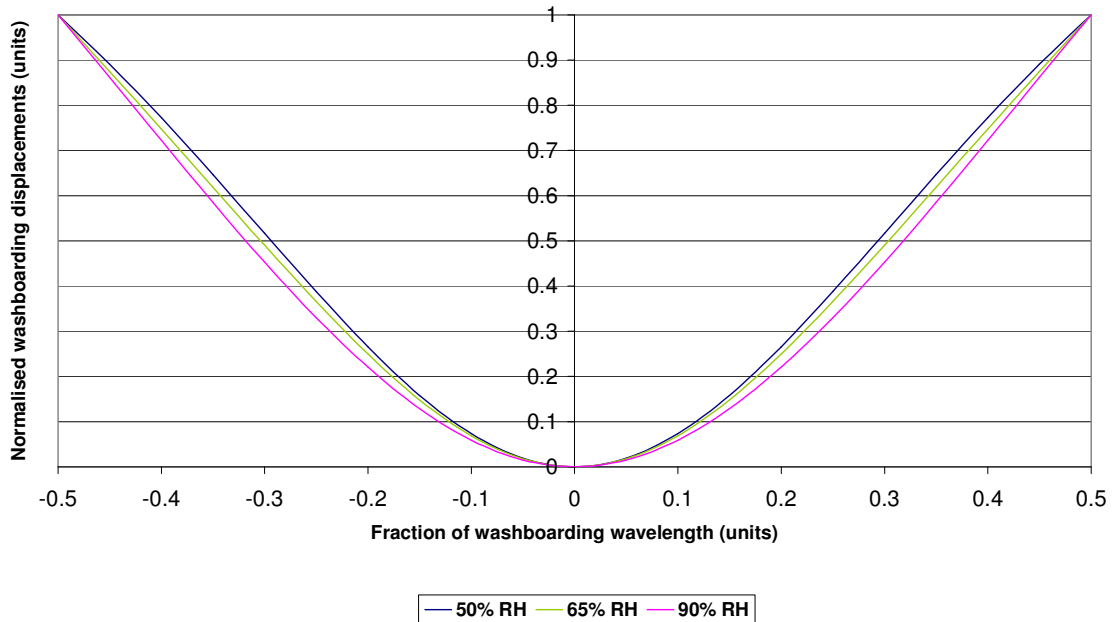


Figure [6.3.3] – The geometry of washboarding after five humidity cycles (day 5) for the sample at 50, 65, and 90 % relative humidities.

Figure [6.3.4] shows the relatively little change in shape after six days of cyclic humidity.

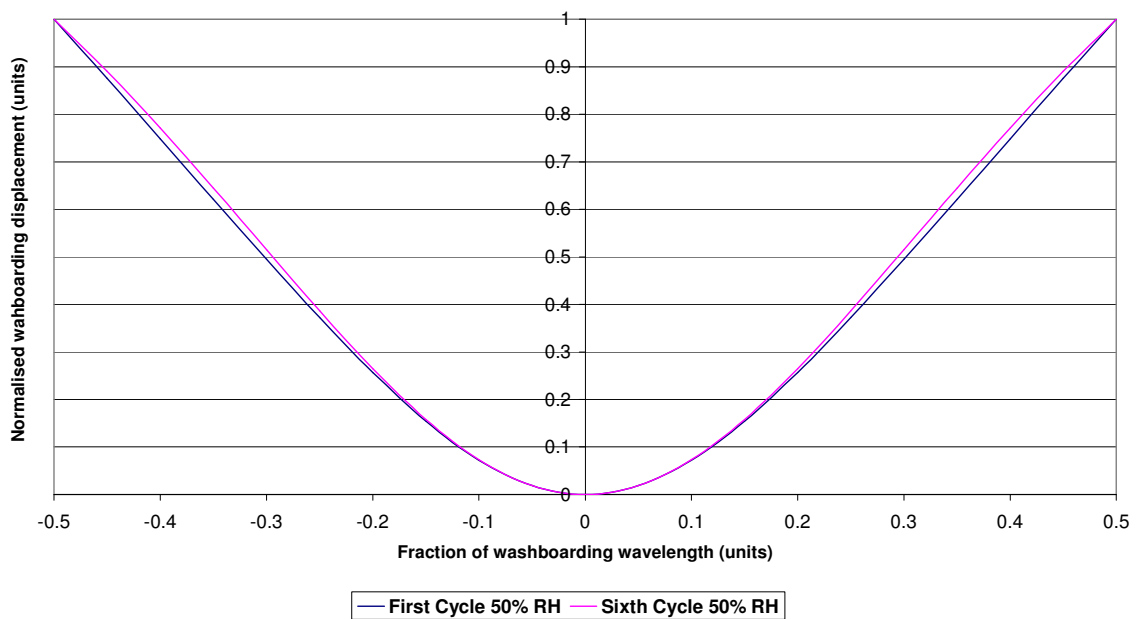


Figure [6.3.4] – Shape of washboarding sampled at the first cycle (day 1) and at the sixth cycle (day 6) of relative humidity.

As the relative humidity changed from 50 % to 90 %, the shape of the sample deviated, on average, by 4 %. This is relatively small in comparison to the change seen in the washboarding depth of 60 % when relative humidity changes over the same range. The average deviation was calculated by taking the difference of the two shapes and averaging the difference over the profile. It is apparent from these curves that the changes in shape are relatively small compared to the change in washboarding depth.

6.3.2 Finite element analysis (FEA) study of moisture in corrugated board

The properties of paper and glue change with moisture content and hence relative humidity of the papers' environment. For this study, the paper's stress and strain characteristics, as well as the change in thickness (due to hygroexpansion) were set to correspond to the relative humidities of 23, 50, 78 and 90 %. The simulated glue force (see section [5.1.2]) was reduced by a factor related to the decrease in elastic modulus as moisture increased (see section [3.2] for details). The results, including a summary of paper properties and simulated glue forces are presented in table [6.3.1].

Table [6.3.1] – Summary of paper properties, glue properties and washboarding results for four different relative humidities using finite element analysis.

Relative Humidity (%)	Simulated Glue Force (%)	Paper Modulus (MPa)	Paper Thickness (mm)	Washboarding (um)
23	100	7060	0.304	50
50	71	6450	0.309	39
78	41	5190	0.320	30
90	28	4070	0.329	25.4

The results are illustrated in figure [6.3.5] below. This figure shows that there is a decrease of washboarding (y) with an increase in humidity (x). The line of best fit corresponds to $y = 57.9 - 0.36x$ with a high R^2 value of greater than 0.99. The decrease in washboarding depth was approximately 50 % from 50 % to 90 % relative humidity for this paper and glue quantity, compared to decrease of 60 % observed experimentally for the cyclic investigation in section

[6.3.1]. This is in reasonable agreement considering that the paper mechanical properties may not have matched the FEA model.

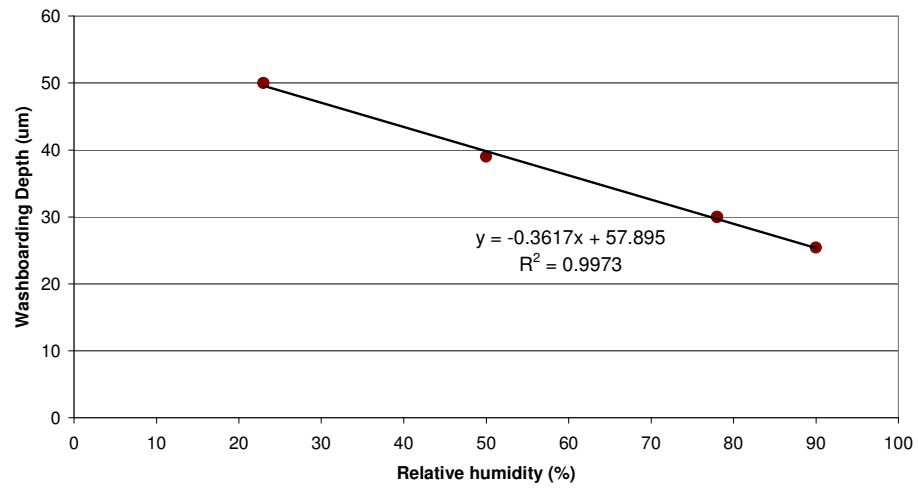


Figure [6.3.5] – Relationship between relative humidity and washboard depths using finite element analysis

6.4. The effect of washboarding upon the performance of corrugated cardboard

6.4.1 Edge crush test performance

6.4.1(a) Empirical testing

A range of corrugated boards were tested for edge crush performance, see section [5.4.1(a)] for details. The results of the study are summarised in table [6.4.1].

Table [6.4.1] – Results of edge crush test (ECT) study. *Standard deviation of eight repetitions.

Sample #	Fluting Type	Weight								ECT		Washboarding Depth		
		Total		Single Facer		Double Backer		Fluting		Mean (N)	SD* (N)	Front (um)	Back (um)	Mean (um)
		(g)	(g/m ²)	(g)	(g/m ²)	(g)	(g/m ²)	(g)	(g/m ²)					
F01	c	5.45	545	1.45	145	1.65	165	2.16	150	723	20	40	70	55
F02	c	5.26	526	1.53	153	1.63	163	1.84	128	745	13	98	60	79
F03	c	5.46	546	1.42	142	1.51	151	2.30	160	692	27	48	109	78
F04	c	5.10	510	1.75	175	1.76	176	1.56	108	704	23	36	93	64
F05	c	6.64	664	2.27	227	2.00	200	2.17	151	919	28	65	84	74
F06	c	6.66	666	2.11	211	2.18	218	2.18	151	991	32	79	24	52
F07	c	6.28	628	2.18	218	2.13	213	2.00	139	869	31	84	31	58
F08	c	7.43	743	2.54	254	2.52	252	2.38	165	1034	28	53	17	35
F09	c	6.32	632	2.12	212	2.14	214	1.88	131	812	31	55	24	39
F10	a	7.20	720	2.55	255	2.58	258	1.98	129	899	31	65	29	47
F11	a	5.20	520	1.55	155	1.52	152	2.00	130	673	21	43	18	31
F12	e	8.34	834	2.99	299	3.01	301	2.10	164	1199	47	33	16	24

The washboarding depth was measured for both the single facer and double backer sides. A good correlation was noticed between the average washboarding depth of both sides and the edge crush test (ECT) failure load. This is illustrated in figures [6.4.1] and [6.4.2].

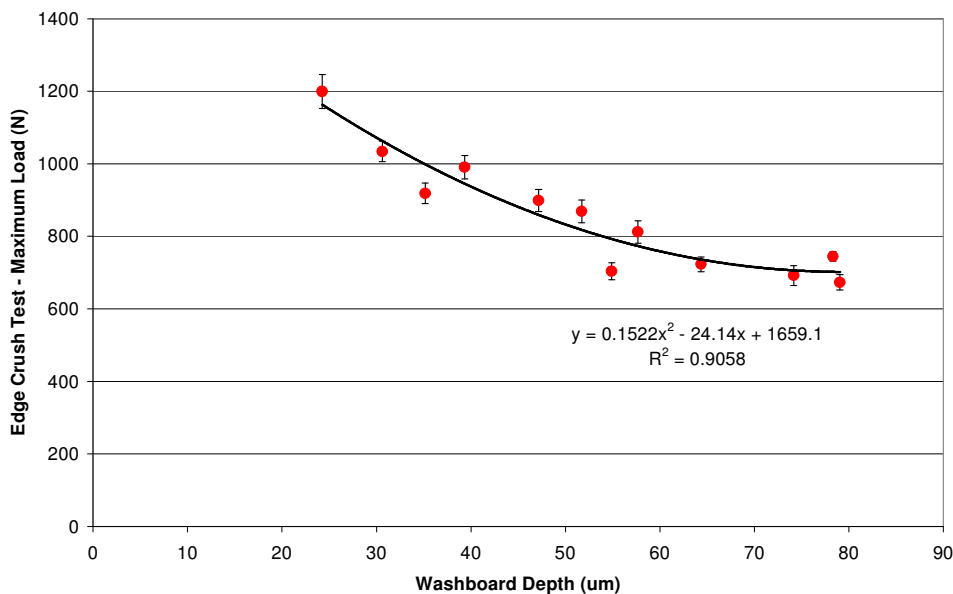


Figure [6.4.1] – Relationship between washboarding depth and ECT failure load.

Figure [6.4.1] depicts washboarding depth in μm (x) versus ECT in Newtons (y). A line of

best fit was found that fit the data well ($y = 0.152 \cdot x^2 + 24.1 \cdot x + 1660$) with a correlation, R^2 value of 0.91.

With an increase in the depth of washboarding there is a decrease observed in the ECT loads required to cause a sample to fail. This does not necessarily mean that a washboarding depth is the main factor in ECT response, as other parameters such as an increase in total board grammage is also highly correlated to an increase in ECT values.

Figure [6.4.2] shows that the relationship between board grammage (x) and ECT (y) is highly correlated with a linear relationship given by $y = 1.37 \cdot x$ and a R^2 value of 0.92.

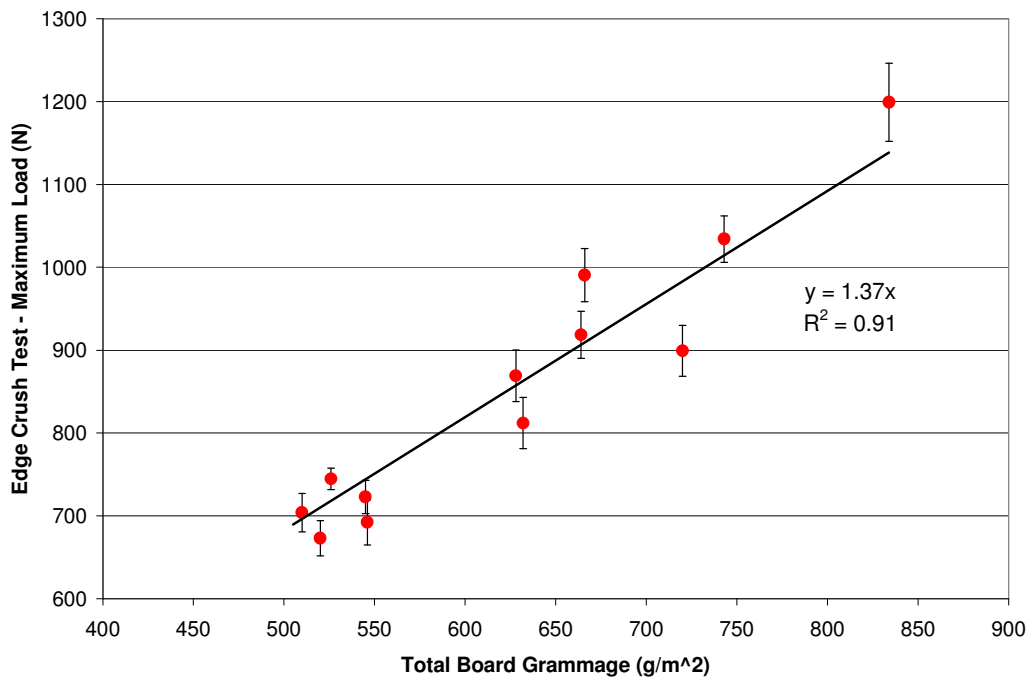


Figure [6.4.2] – Relationship between total board grammage and ECT failure load.

These correlations may be expected because an increase in board grammage will decrease washboard depth. Figure [6.4.3] shows the relationship between grammage (x) and washboarding depth (y) for the range of samples whose properties are shown in table [6.4.1]. The line of best fit is given by the linear equation, $y = 152 - 0.159 \cdot x$ for the sample tested.

This time, however, the correlation is lower with a R^2 value of 0.77. This lower value is likely due to the variation in glue quantities, paper stiffness and thickness as well as the mix of different flute types for the different cardboard.

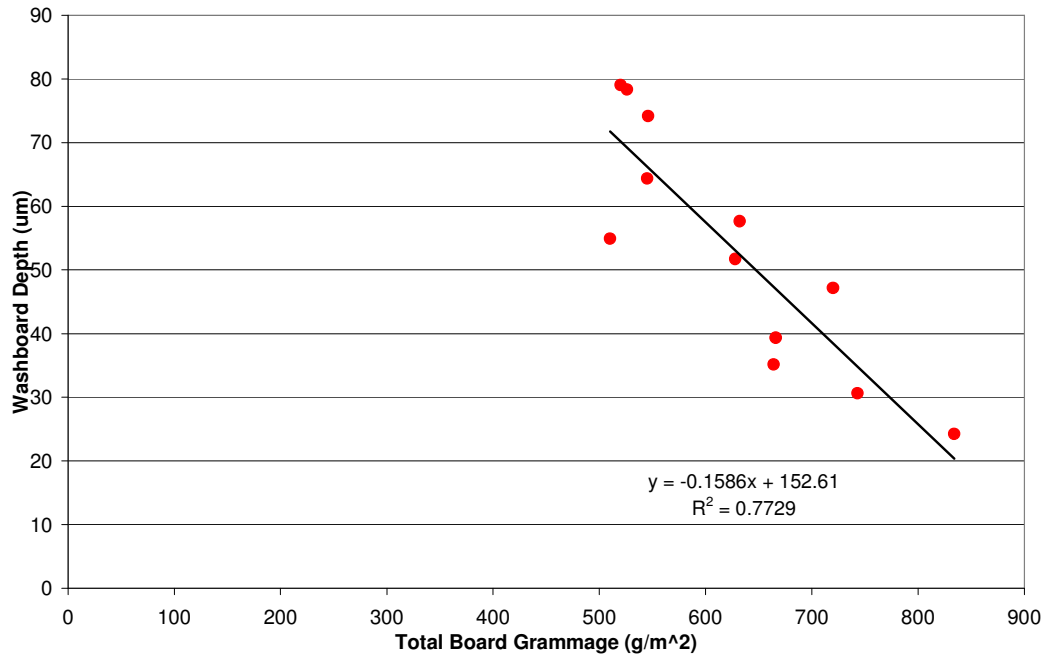


Figure [6.4.3] – Relationship between board grammage and washboarding depth.

Washboarding depth, board grammage and ECT failure loads are not independent, therefore making it difficult to conclude which factor is most important. Finite element analysis was used to attempt to determine how these factors contribute to ECT failure loads.

6.4.1(b) Finite element analysis of the edge crush test

Figures [6.4.4] and [6.4.5] show the compression for an ECT model, where figure [6.4.4] shows a colour map of the y-displacement and figure [6.4.5] shows a stress colour map. Figure [6.4.5] indicates that stress concentrates at the edges of the sample. This is the region where the samples fail both in reality and in this model.

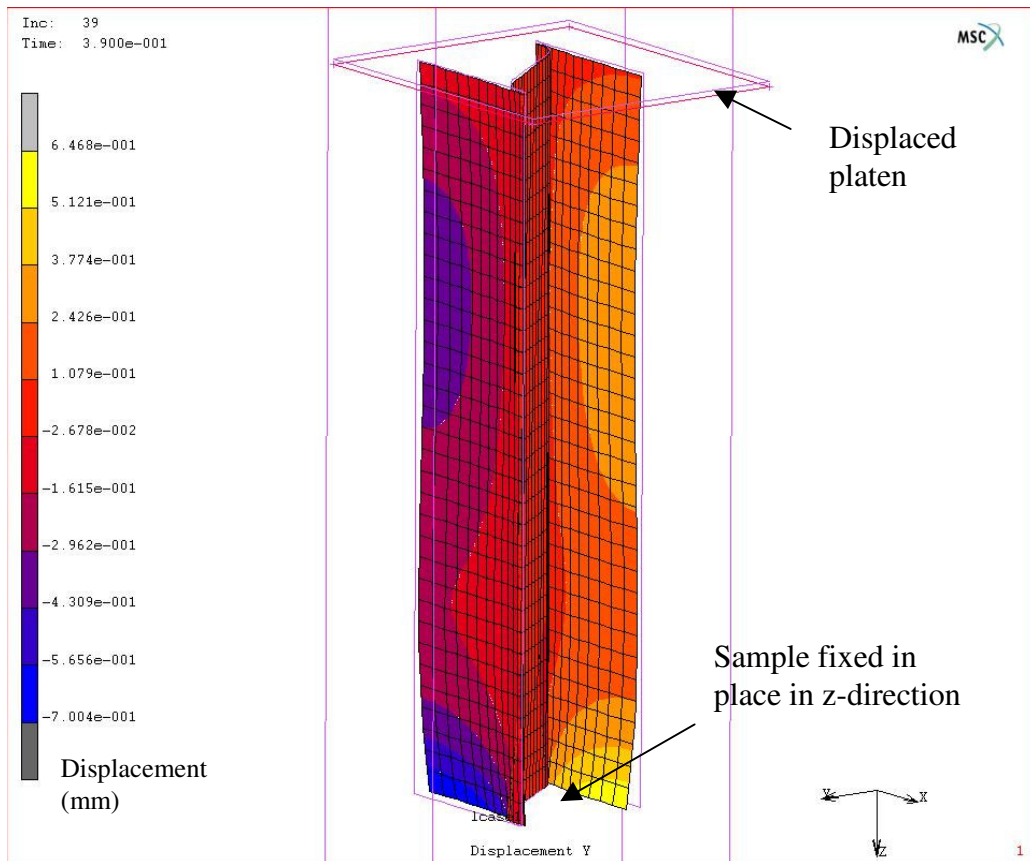


Figure [6.4.4] – A y-displacement colour map of a finite element model depicting an ECT sample under load

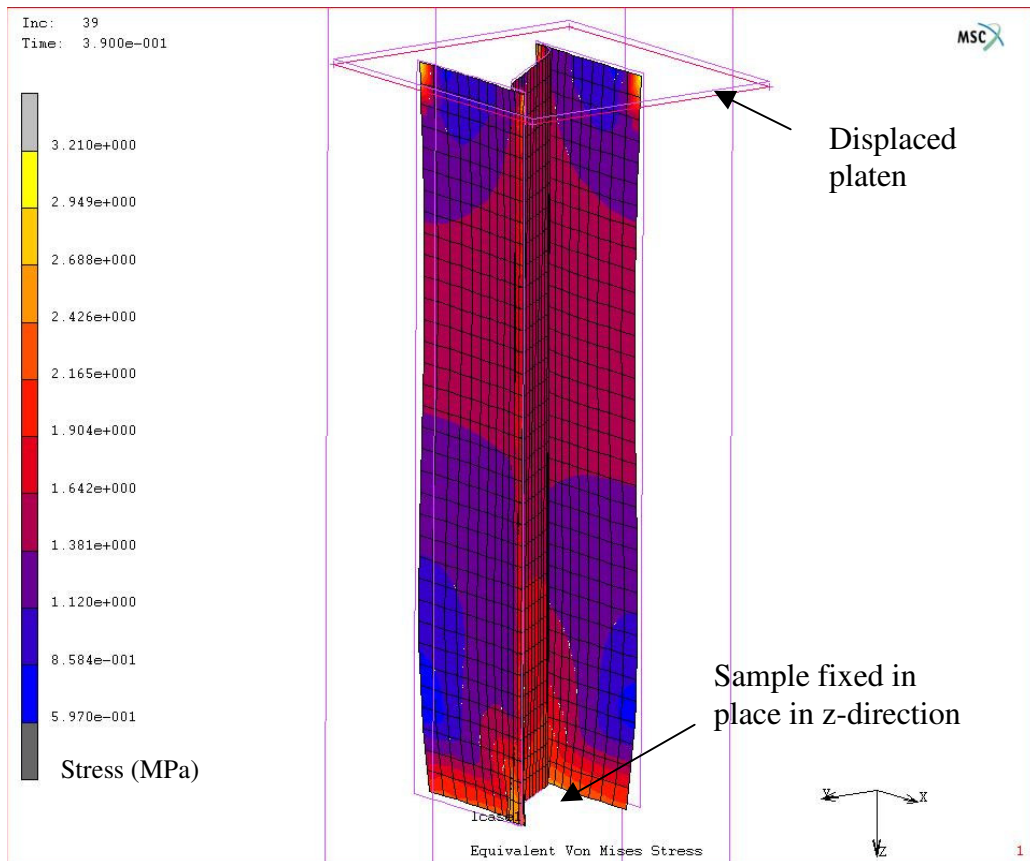


Figure [6.4.5] – A stress colour map of a finite element model depicting an ECT sample under load

Sixteen models were evaluated, using four simulated glue forces strengths and four different paper types. The properties of the paper were varied in stiffness and failure load (STFI) to represent a typical range. Table [6.4.2] shows a summary of the results, including the paper properties used. A simulated glue force was chosen to create a typical range of washboarding depths seen experimentally, ranging from 35 to 65 μm . The simulated glue force needed to create this range of washboarding was then designated as 100 %, as seen in table [6.4.2]. All other simulated glue forces applied are then relative to this force.

Table [6.4.2] – Parameters used in the sixteen finite element models to investigate ECT. Also shown are the resulting ECT failure loads and washboarding depths produced.

Glue Force %	Modulus (MPa)	STFI (kN/m)	ECT (N)	Washboarding Depth μm
37.5	7060	4.20	796	4.62
37.5	6450	3.50	743	4.09
37.5	5800	3.00	707	4.01
37.5	5190	2.52	694	4.47
75.0	7060	4.20	721	19.6
75.0	6450	3.50	661	19.2
75.0	5800	3.00	622	20.6
75.0	5190	2.52	615	24.4
100	7060	4.20	643	35.0
100	6450	3.50	580	36.0
100	5800	3.00	547	42.0
100	5190	2.52	545	65.3
110	7060	4.20	547	42.6
110	6450	3.50	477	46.8
110	5800	3.00	464	60.4
110	5190	2.52	464	168

The results in table [6.4.2] were mapped and are shown in figure [6.4.6]. From this figure it is evident that both the stiffness of the paper and the washboarding severity have an effect on the final ECT failure load. For example, an increase in paper elastic modulus from 5190 to 7060 MPa relates to ECT failure loads ranging from 550N to 800N, for a simulated glue force of 37 % which corresponds to a very low washboarding depth (approximately 4 μm).

For each of the four different paper models, an increase in washboarding decreased the load needed to fail an ECT sample, eg. if the washboarding depth is decreased from 50 to 5 μm , then the ECT failure load is increased from 700 N to 800 N, for a paper with a modulus of 7060 MPa.

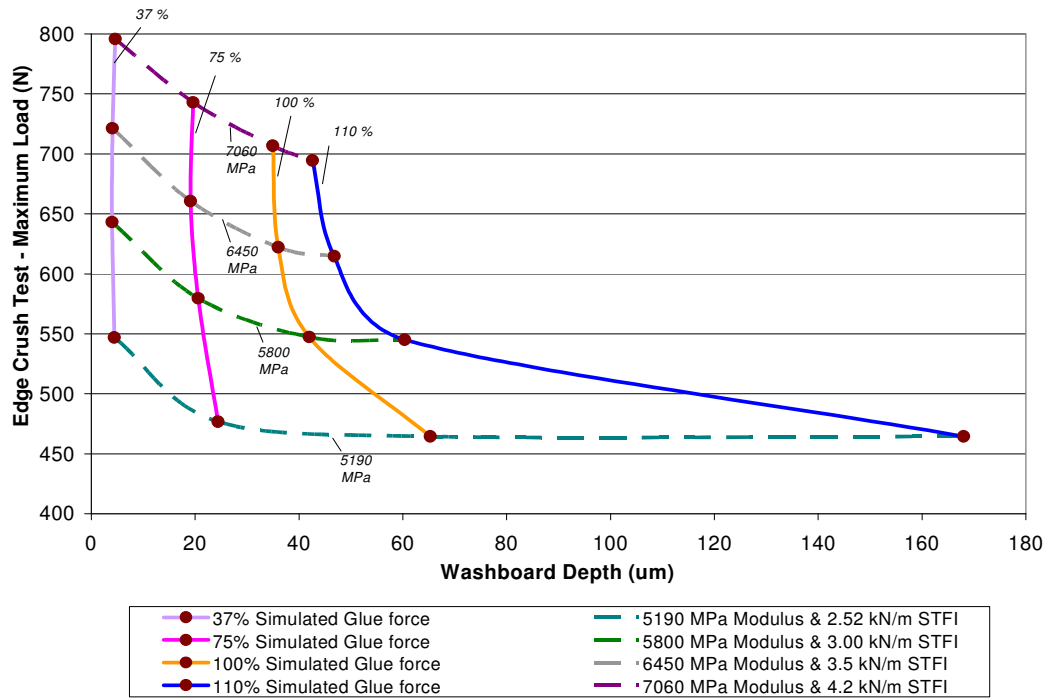


Figure [6.4.6] - Mapping of washboard depth, ECT failure load, percentage simulated glue force and paper properties.

Decreasing the glue pressure and increasing the paper stiffness raised the ECT failure load, as illustrated in figure [6.4.7].

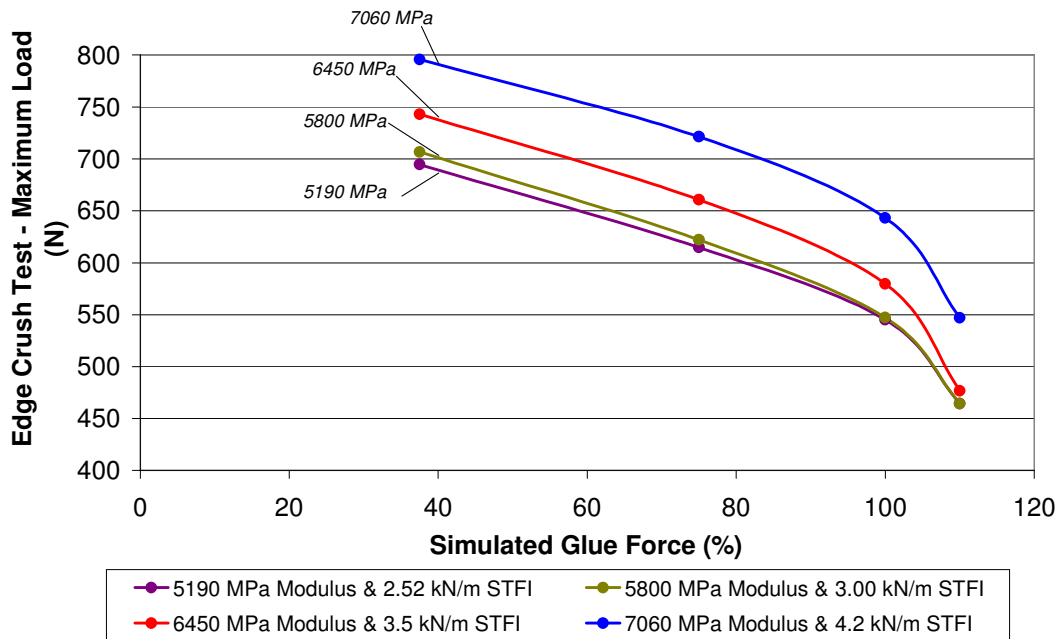


Figure [6.4.7] – Relationship between simulated glue force, paper elastic modulus and ECT failure load.

Similar to the results shown empirically in section [6.4.1(a)], the results presented in this section show that both board weight and washboarding affect the ECT performance of corrugated board.

It is difficult to ascertain the relative influence of grammage (seen as an increase in modulus and STFI failure) of paper and of washboarding on ECT failure load from the FEA results. However it appears that an increase in washboard depth of approximately 300 % observed in figure [6.4.6] showed a similar drop in ECT failure load, as did a decrease of approximately 36 % in paper modulus or 40 % STFI. This relative percentage change may indicate that grammage of the paper has more influence on ECT failure load.

Overall, FEA modelling showed that an increase in washboarding decreased ECT, while an increase in paper stiffness increased it.

6.4.2 MD-Shear performance

6.4.2(a) Empirical testing

MD-Shear testing was performed on the same samples as used for the three-point bend testing. Table [6.4.3] shows a summary of the results. See section [4.4.2(a)] for details regarding how MD-Shear measurements were performed. The paper basis weight and composition of the papers used are shown in table [5.4.1].

Table [6.4.3] – Results for MD-Shear study. *Standard deviation of eight repetitions.

Washboarding Depth					MD-Shear	
Single Facer		Double Backer		Combination	Mean (N/m)	SD* (N/m)
Mean (um)	SD* (um)	Mean (um)	SD* (um)	Mean (um)		
34	5.7	15	1.0	25	16.19	0.256
61	6.3	18	1.8	39	17.37	0.221
33	5.5	19	0.4	26	16.19	0.277
42	4.9	27	0.8	34	16.88	0.147

The relationship between the washboarding depth and MD-Shear values is illustrated in figure [6.4.8], demonstrating a clear relationship between the two. An increase of 7 % in MD-Shear value was seen at a washboarding depth of 40 μm compared to that seen at the depth of 25 μm .

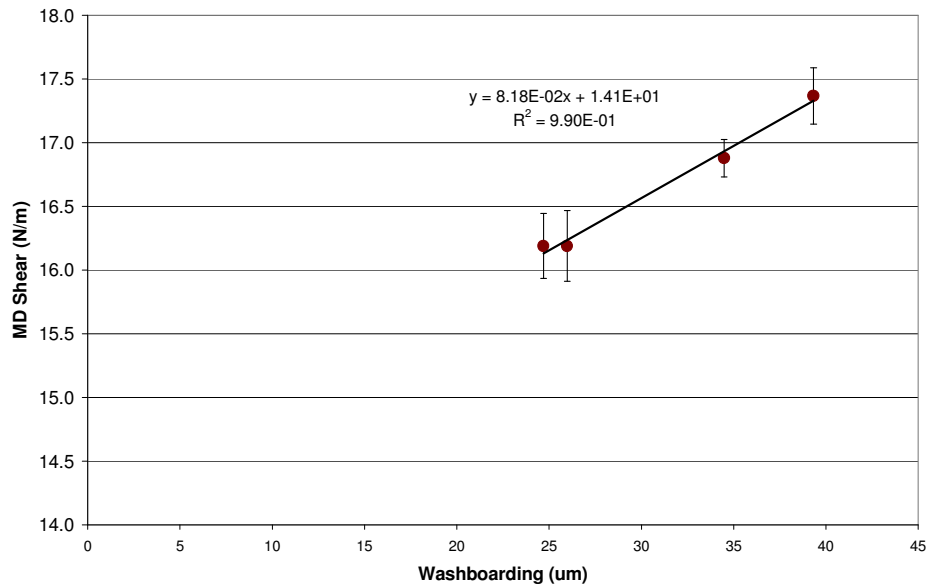


Figure [6.4.8] – Relationship between washboarding depth and MD-Shear values.

6.4.2(b) Finite element analysis of MD-Shear

Finite element analysis was used to see if the above empirical results could be confirmed (see method section [5.4.2(b)] for details). The paper properties used were an elastic modulus of 6450 MPa, a thickness of 310 μm and a compression failure of 3.5 kN/m (STFI). A change in simulated glue force was modelled corresponding to the changes in washboarding depth.

A model undergoing the twist displacement for the MD-Shear calculation is seen in figure [6.4.9], with a colour map showing the y-deflection.

The models differed in the amount of simulated glue force applied, therefore changing the amount of washboarding present. Figure [6.4.10] illustrates the results corresponding to

washboarding depths of 0, 4.6, 17 and 40 μm while figure [6.4.11] shows the calculated MD-Shear values from the graphs in figure [6.4.10].

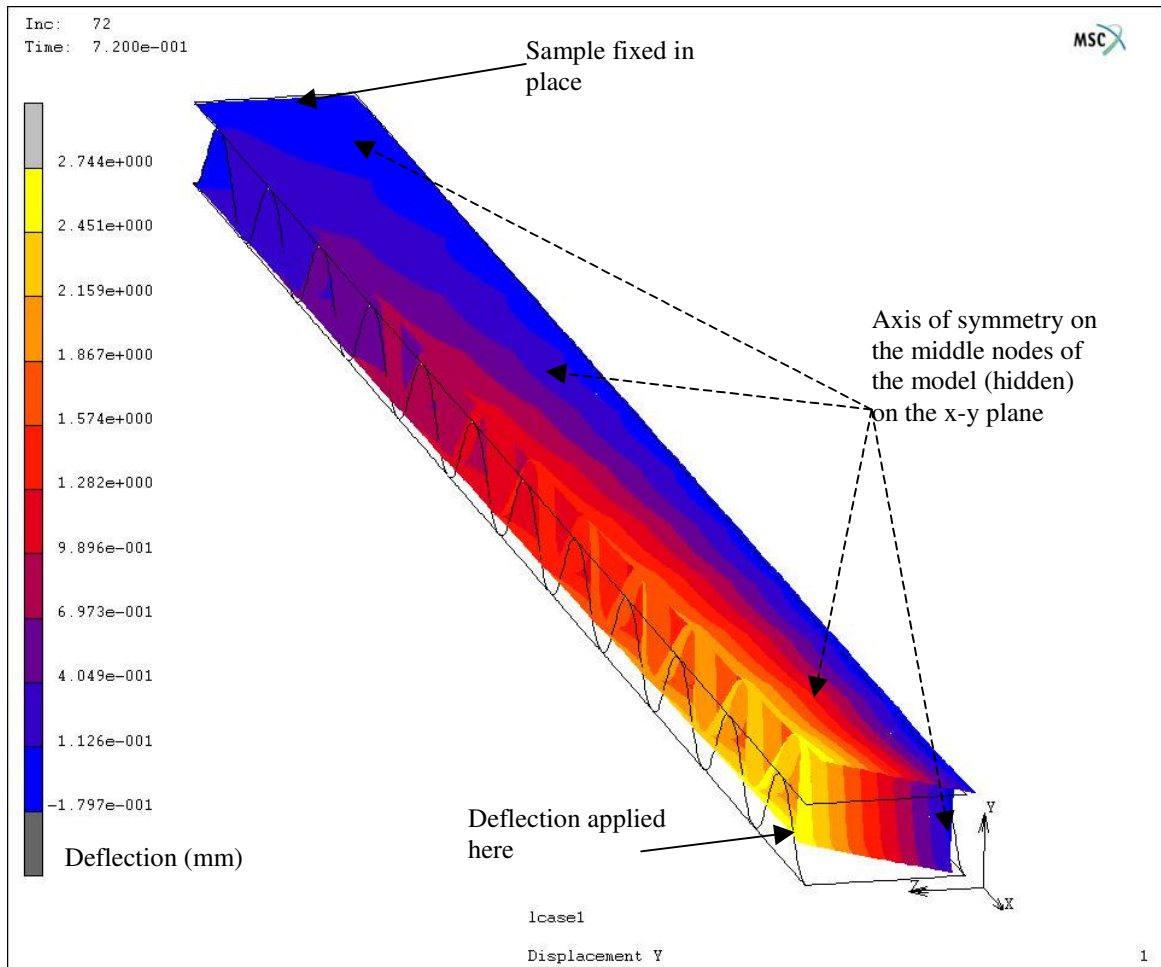


Figure [6.4.9] – A y-displacement colour map of a finite element model of a MD-Shear sample undergoing testing.

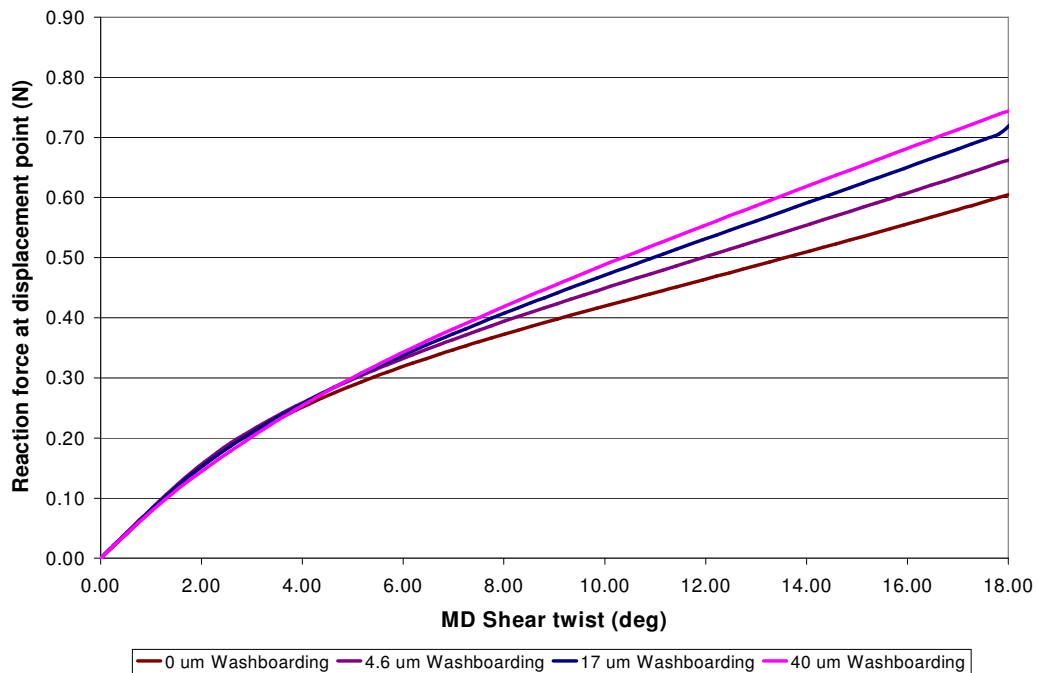


Figure [6.4.10] – MD-Shear load versus rotation graphs for a range of washboarding depths using finite element analysis.

Included in figure [6.4.11] is a MD-Shear value for a washboarding depth of 133 μm , the curve for this depth was not included in figure [6.4.10]. This amount of washboarding depth placed the paper in its plastic deformation region. The twist was calculated from the displacement of the node shown in figure [6.4.9] and the width of the sample.

The graph in figure [6.4.11] shows that an increase in washboarding increased the MD-Shear value until the washboard depth exceeded 50 μm . The MD-Shear value at 40 μm was approximately 3 % higher than at 25 μm , which was less than half of that seen for the physical testing (7 % increase over the corresponding range). The reason for this difference is not clear but may be explained by the different sample sizes used in the empirical testing and the computer limitations of the FEA model (refer to section [5.4.2(b)] for details). It may also be due to the isotropic modelling of the paper, where machine manufactured paper is anisotropic [48, 49, 51]. The MD-Shear test has components in the CD due to twisting of the sample, so it is unclear what affect anisotropic modelling would have on this result. The FEA

results showed an increase in the MD-Shear value of 22 % with a corresponding increase in washboarding depth from 0 to 40 μm .

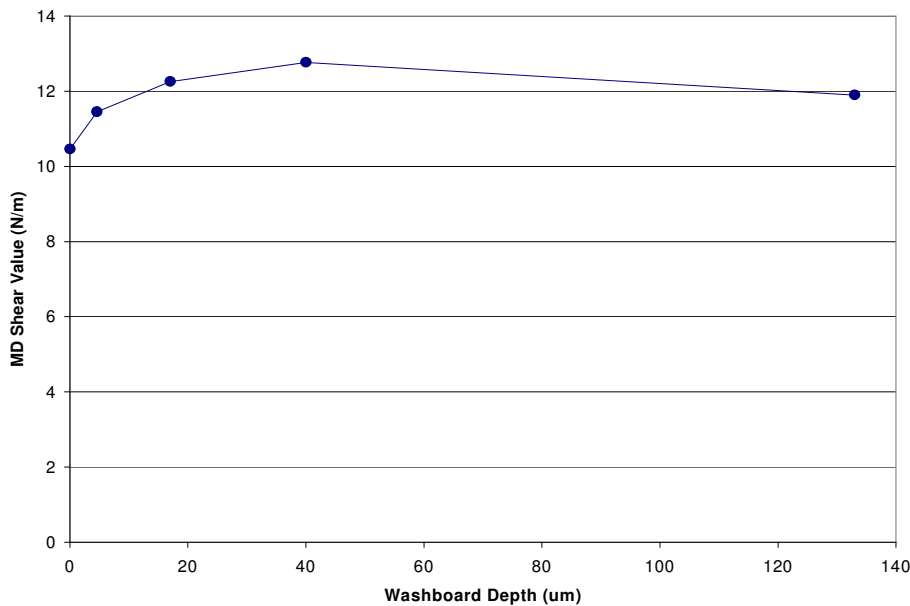


Figure [6.4.11] – Relationship between washboarding depth and MD-Shear values.

When the liner entered the plastic deformation region for a washboarding depth greater than 40 μm , the MD-Shear value dropped. This suggests that there is an ideal washboard depth for maximum MD-Shear performance for a corrugated board. The corrugated board samples used in the physical testing did not display the decrease in MD-Shear seen in figure [6.4.11]. This may have been because the strain (seen as washboarding) was not large enough for the liner to be in the non-linear region of the stress-strain curve.

6.4.3 Three-point bend test performance

6.4.3(a) Empirical testing

Three-point testing was performed on C-flute samples, where the washboarding depth was measured and three-point bend failure load noted, as it was the only measurement that changed significantly between samples (see section [5.4.3]). Results are shown in table [6.4.4], and graphed in figure [6.4.12]. The washboard depths were averaged between single facer and double backer sides, and the three-point bend tests were performed in two orientations, single facer up and single facer down, with the failure loads averaged. If only

one test direction were used it would likely bias the test as there is a variation in washboard depths on the two faces. The difference may be significant considering that paper performs differently in tension and compression [89]. The paper basis weight and composition of the papers used are shown in table [5.4.1].

Table [6.4.4] - Data of three-point test study. *Standard deviation of eight repetitions

Washboarding Depth					Failure Load	
Single Facer		Double Backer		Combination	Both Directions	
Mean	SD*	Mean	SD*	Mean	Mean	SD*
(um)	(um)	(um)	(um)	(um)	(N)	(N)
34	5.7	15	1.0	25	2.17	0.113
61	6.3	18	1.8	39	2.24	0.124
33	5.5	19	0.4	26	2.15	0.078
42	4.9	27	0.8	34	2.18	0.111

The results show that there is a slight increase in failure load for an increase in washboarding depth. The errors bars (one standard deviation) show how the failure load is different in the two directions. Even though the error bars are large the correlation is relatively good with a R² value of 0.77.

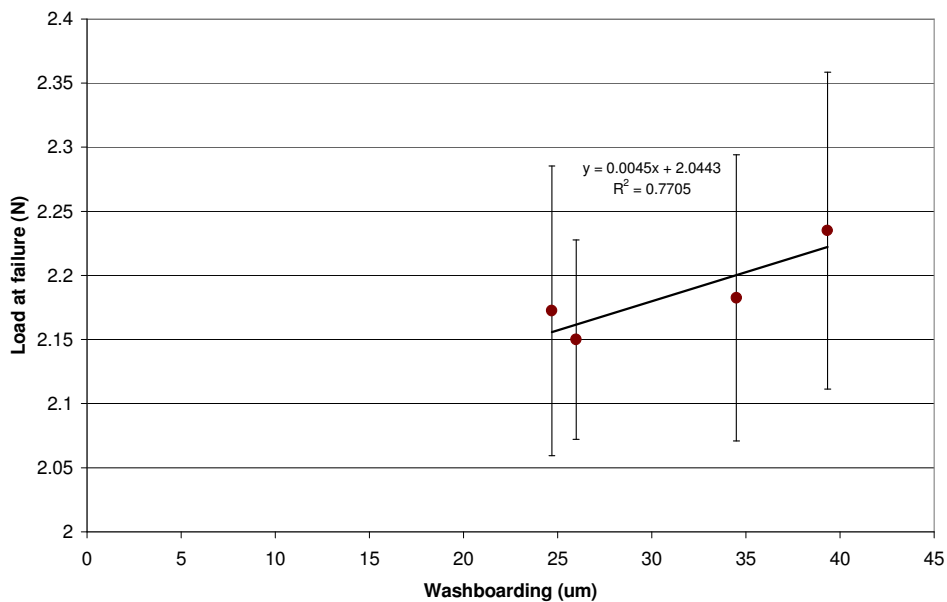


Figure [6.4.12] – Relationship between washboarding depth and load at failure.

However, a doubling in washboard depth changes failure load by only 3 % and is therefore relatively insignificant considering the large standard deviations of the experimental data as shown in the figure above.

6.4.3(b) Finite element analysis of three-point bend testing

Finite element analysis was used to see at what point the washboarding depth could become significant for the three-point test, and to see if it confirms the small increase in failure load with increasing washboarding depth. The properties of the paper used were an elastic modulus of 6450 MPa with a thickness of 309 μm and a compression failure of 3.5 kN/m (STFI). Figure [6.4.13] shows a colour map of the stress distribution of the model prior to three-point testing but after applying the simulated glue force.

Figures [6.4.14] through to [6.4.16] show the point of failure of the sample under load where figures [6.4.14] and [6.4.15] show the deflections and figure [6.4.16] shows the deflection with the corresponding stress colour map.

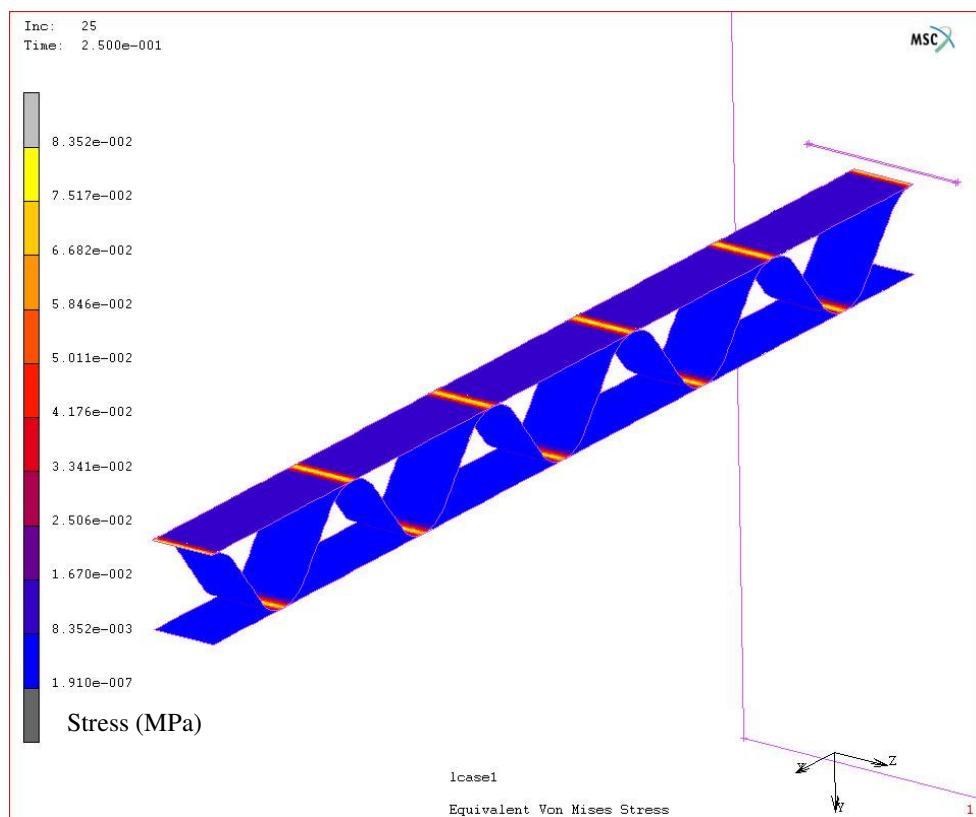


Figure [6.4.13] – A colour stress map of a three-point bend model after the application of simulated glue force.

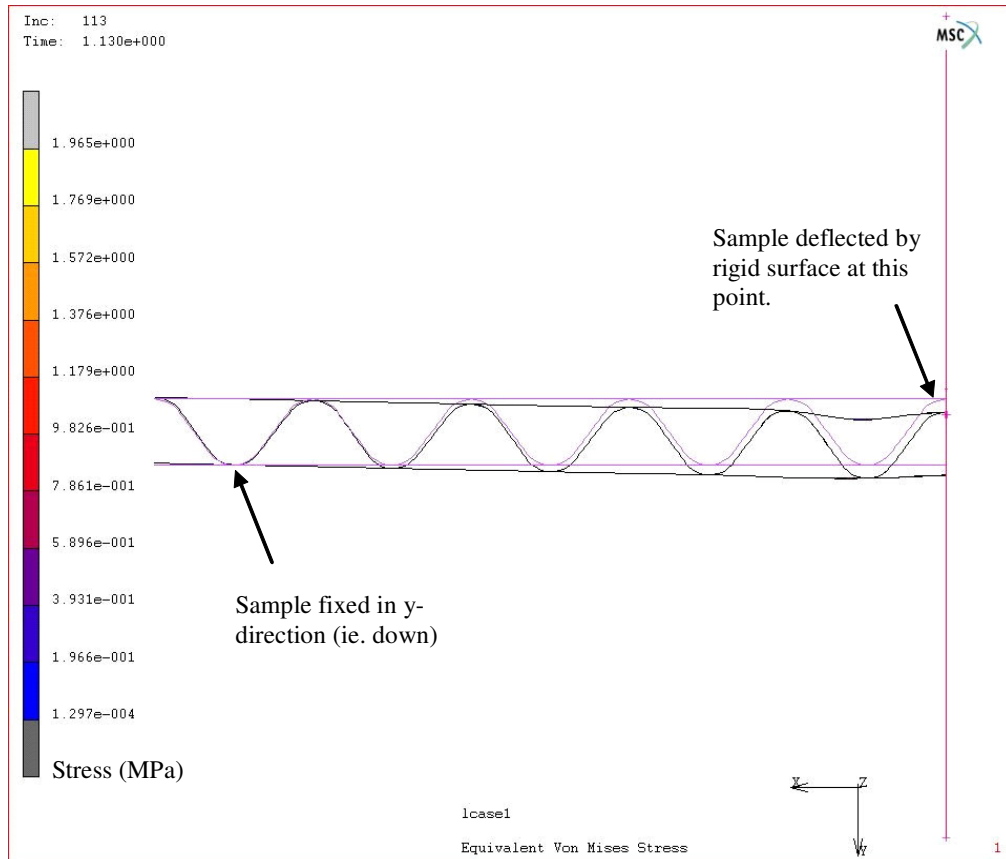


Figure [6.4.14] – Side on view of finite element model depicting a three-point test at failure.

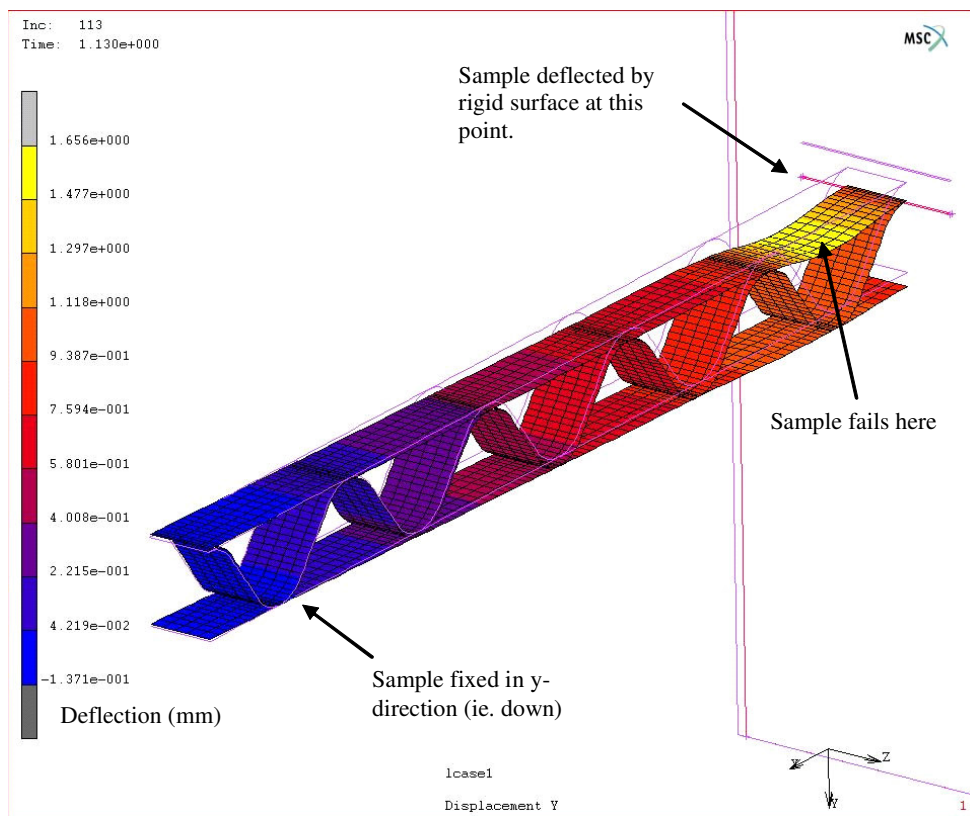


Figure [6.4.15] – Finite element model depicting a y-displacement colour map of a three-point test at failure.

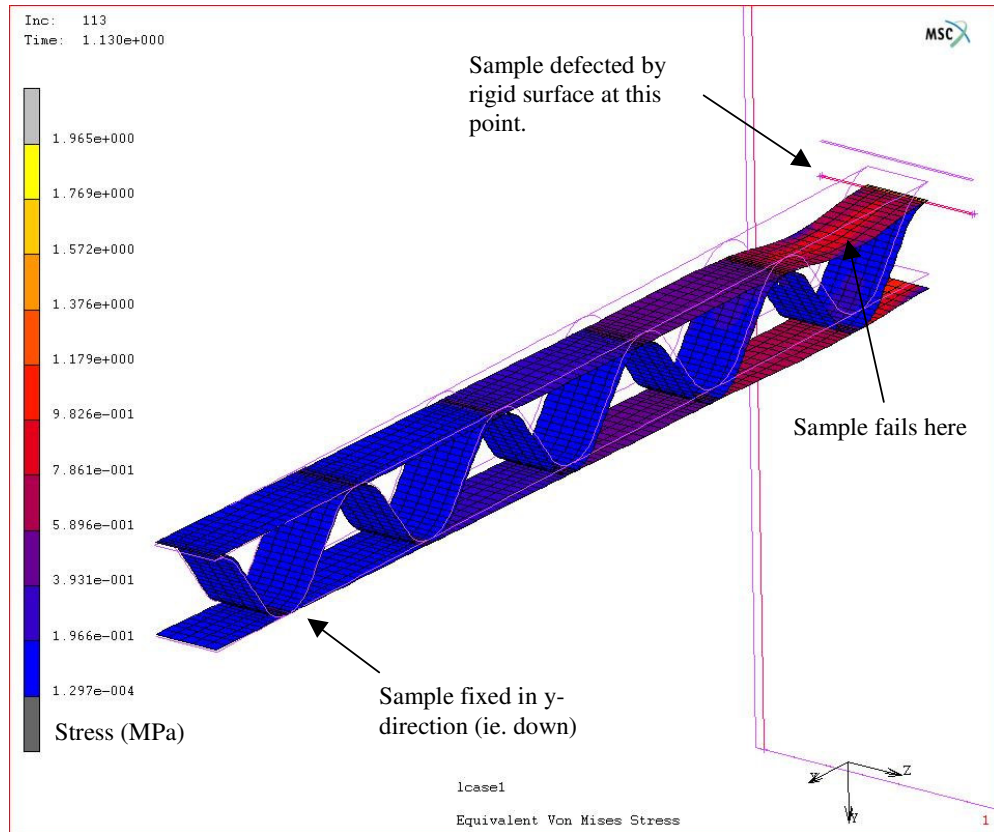


Figure [6.4.16] – Finite element model depicting a stress colour map of a three-point test at failure.

The load versus deflection graphs for the different glue pressures (and therefore washboarding depths) is presented in figure [6.4.17]. The sample was deflected and the reaction force (the force necessary to bend the sample to this point) was calculated by FEA. This figure displays a reduction in bending stiffness, possibly due to a reduction in sample thickness (flute tip to tip), similar to that found from FEA of washboarding presented in section [6.2.6]. As the deflection of the sample was increased, the resistance to bending, measured as a force, increased until a point where the three-point bend sample began to fail. At this point any further increase in deflection was met by less resistance. This appeared as the knee points in figure [6.4.17].

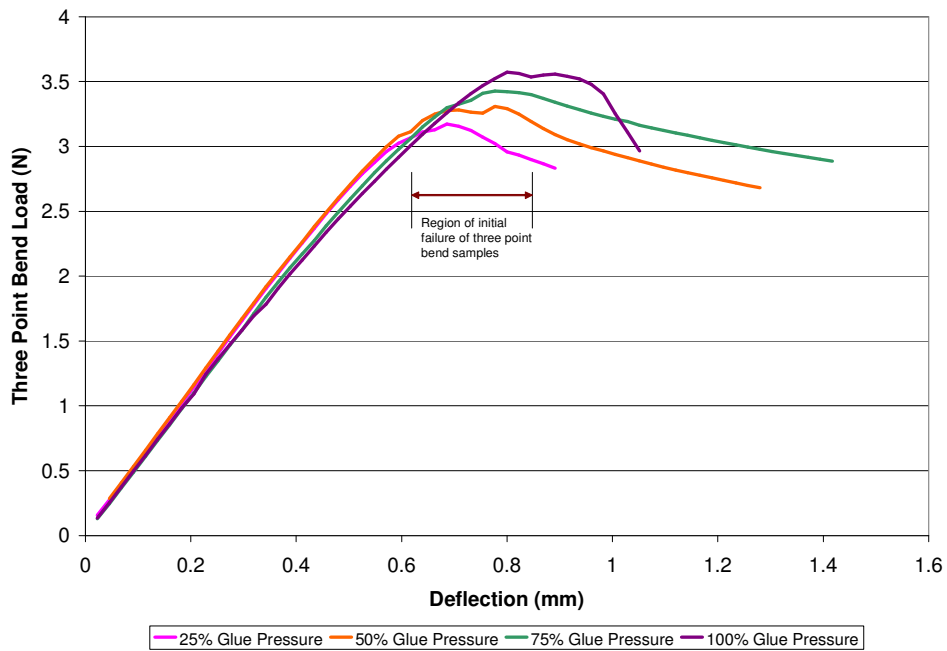


Figure [6.4.17] – Load versus displacement graphs for a range of glue forces (and therefore washboarding) calculated using finite element analysis.

An increase in failure load is shown in figure [6.4.18] for an increase in washboarding depth. It is in accordance with the result from the physical testing that for a doubling in washboarding depth, the equivalent increase in three-point bend load at failure was only ~3 % higher.

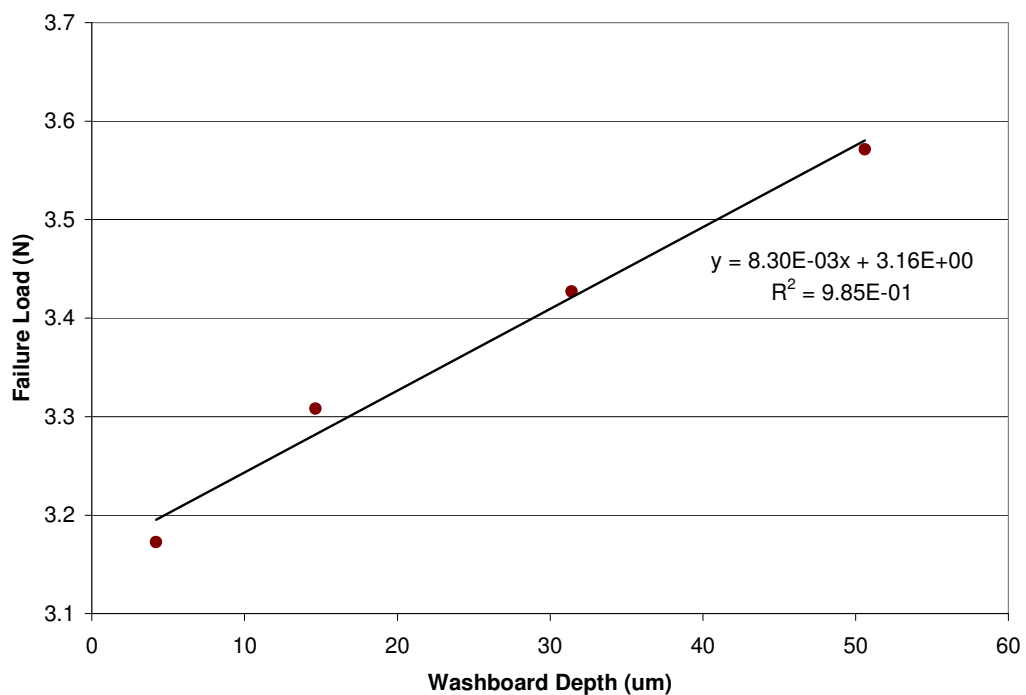


Figure [6.4.18] – Relationship between washboarding depth and three-point bend failure.

It is important to keep in mind the paper modelled in the FEA was isotropic, compared to machine manufactured paper, which is anisotropic in nature [48, 49, 51]. However, it is not expected that the results would differ much if anisotropic calculations were performed due to the little force components (except in the machine direction), other than the Poisson effect [53].

6.4.4 Final box strength and performance measures

The effect of washboarding upon the three performance measures, consisting of edge crush, three-point bending and MD-Shear tests differ. For increasing washboarding, the edge crush test showed a lowering of performance, while the three-point bend displayed relatively little change in failure load, and MD-Shear showed an increase in performance. How much each of these performance measures relates to the final box strength is unclear as these tests have different relevance depending on the packaging application [15], but these results show clearly that washboarding depth can influence the strength of corrugated board.

Due to the proprietary nature of MD-Shear, there is little published information in comparison to internal knowledge, apart from a paper by Selway and Kirkpatrick [84]. Because of this, Russell Allan, an expert within Amcor regarding MD-Shear, was consulted regarding the above findings. The following paragraphs (in *italics*) are his analysis of the results.

The increase in MD-Shear values for an increase in washboarding is possible because the MD-Shear test only reflects, to a large extent, the medium performance and ignores the liner effect. The additional glue that causes greater washboard levels presumably locks the medium shape in place (by wedging it at the top of the flutes) and this may serve to increase the MD-Shear or shear rigidity of the medium structure. Typically one expects ECT to most closely follow the ultimate box compression performance observed. Below an MD-Shear value of

around 20 kN/m the MD-Shear effect will be one of degradation of the potential box compression performance.

So, it is likely that the overall effect of the washboarding upon the box performance is marginal if the MD-Shear levels are below 20 kN/m. ie. the decrease in ECT may be compensated by an increase in the MD-Shear performance of the box.

6.5 The effect of washboarding upon the printing quality of corrugated cardboard

6.5.1 Full-tone coverage

Corrugator manufactured board samples were chosen with different furnishes for an investigation of full-tone printability for a specific pressure during printing (see section [5.5] for details). The printing conditions used are provided in section [5.5.2]. The washboard depth and print stripe coverage were measured as described in section [3.5]. The results of this investigation are shown in table [6.5.1]

Table [6.5.1] - Results of washboard depth and print coverage study.

Liner colour	Washboarding Depth (um)	- Paint stripe ratio (%)			
		Sample #1	Sample #2	Mean	Standard deviation
white	43.5	13.0	15.0	14.0	1.0
white	45.9	11.5	11.6	11.6	0.0
brown	24.3	2.5	2.7	2.6	0.1
white	41.1	3.0	3.5	3.3	0.3
brown	30.7	3.0	2.6	2.8	0.2
brown	111.2	15.8	11.0	13.4	2.4
brown	76.5	10.0	10.0	10.0	0.0
brown	59.7	8.6	6.3	7.5	1.2
brown	52.3	5.0	5.0	5.0	0.0
brown	68.6	9.5	8.0	8.8	0.8
white	21.1	5.0	4.1	4.6	0.5
white	20.5	4.0	2.3	3.2	0.9
white	13.3	3.7	4.7	4.2	0.5

The relationship between washboard depth and stripe severity on white liner is illustrated in figure [6.5.1], while figure [6.5.2] shows a similar relationship, in this case with brown liner.

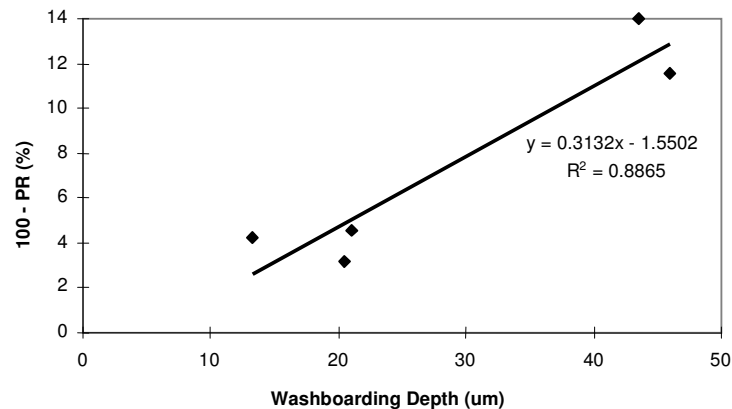


Figure [6.5.1] - Percentage area of no print coverage for a given washboarding depth and constant printing pressure on bleached white liners.

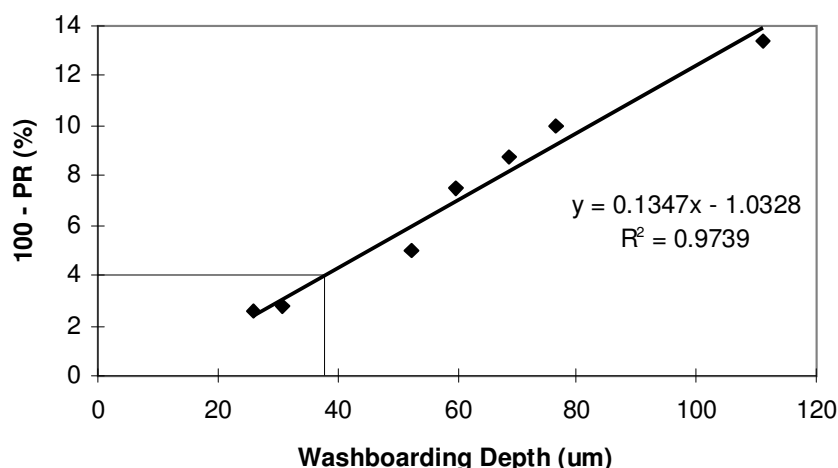


Figure [6.5.2] - Percentage area of no print coverage for a given washboarding depth and constant printing pressure on unbleached brown liners. Samples with measurement of print ratio greater than 96 % (less than 4 % 100-PR) appear to have no striping as judged by a human eye (see the area defined).

The different gradients of 0.13 and 0.31, for the lines of best fit for the white liner and brown liners respectively, was possibly due to surface differences such as surface roughness and/or porosity. The white liner appeared glossy while the brown liner appeared matt.

Roughness may have reduced stripe severity by effectively filling some of the washboarding valleys. A high porosity paper with protruding fibres may also draw the ink to the surface. On the other hand, the white liner's lack of roughness and possible porosity reduced this action.

For brown liner, washboarding depths of less than 40 μm for the level of printing force used appears to the human eye as having full coverage (ie. no stripes). This corresponds to a PR value of greater than or equal to 96 %. Comparatively, for the same printing pressure on bleached white liner, the washboarding depth needs to be closer to 20 μm for the human eye not to detect stripes. Figure [2.4.1] in section [2.4] shows a printed unbleached board with a washboarding depth of 32 μm . This suggests that for good full tone print for a specific printing pressure the washboard depth should be less than a defined thresh-hold, in this case

40 μm printing conditions as provided in section [5.5.2]. Roughness and/or other paper properties may alter this thresh-hold.

6.5.2 Humidity, printability and structural integrity

Damage due to excessive compression of fluting material significantly reduces final box strength, and therefore its survival time under load [26]. Consequently, it may be advantageous to keep printing pressure to a minimum, and because the washboarding depth largely determines this minimum printing pressure, decreasing the washboarding depth may be of benefit. This may be done, as was shown in this thesis, by increasing the moisture content of corrugated board, where an increase in moisture reduced washboarding depth by up to 50 %. An increase in moisture content may be achieved by either increasing relative humidity or decreasing the temperature of the corrugated board for the same relative humidity (see section [6.3.1]).

Chapter 7 - Conclusions

The successful development of the two-dimensional washboarding profilometer and methods (including calibration) allowed a range of studies to be performed on washboarding of corrugated boxes that were not feasible with equipment available previously.

The primary conclusions were as follows: It was found that

- Washboarding depth was linearly related to the amount of glue applied.
- A change in corrugator manufacturing rate increased the amount of glue applied, which in turn increased the amount of washboarding. When glue quantity was similar for different machine speeds, there was little to no change to washboarding depth, suggesting machine speed by itself does not necessarily change washboarding depth.
- A linear change in washboarding depth resulted in a non-linear change in strain.
- Washboarding increased non-linearly with a decrease in elastic modulus (measured ultrasonically). A large increase in washboarding was seen where the measured sonic modulus of the board liner was less than 5.0 GPa. The finite element analysis of the effect of a change in paper modulus upon washboarding supported the above finding. A paper with a typical amount of glue quantity applied was modelled, for which the washboarding depth was seen to greatly increase when the Young's modulus of the paper was less than 5.5 GPa.
- The shape of washboarding changes with relative humidity, but the change was minor when compared to the change in absolute washboarding depth.
- Washboarding depth was highly dependent on the relative humidity of the environment, where an increase in relative humidity decreased the washboarding depth in a linear fashion. It was previously known that an increase in moisture in paper and glue, due to increasing relative humidity, decreases the elastic modulus of paper and glue, while the

thickness of the paper increases. It was found that the reduction in paper stiffness effectively negated the increase in bending stiffness. This leaves the change in glue force as having the greatest effect upon washboarding depth and the primary mechanism that changes the washboarding depth when the relative humidity changes.

- The washboarding depth creeps over time in a cyclic environment, due to stresses imparted on the liner by the glue.
- An increase in washboarding decreases full-tone print coverage (ie. reduction of print quality). A method was developed to measure full-tone print coverage of corrugated board where image analysis was used to determine the full-tone print coverage applied using a flexographic printer. Two relationships between washboarding depth and coverage for a single printing pressure were found, one for bleached liner and one for unbleached liner. If the washboarding depth is kept low enough, then full coverage should occur for a given printing pressure.

The effect of washboarding upon three performance measures (edge crush test (ECT), three-point bend, and MD-Shear (an Amcor Ltd. proprietary test)) was tested empirically and modelled using finite element analysis (FEA). These performance measures gave an indication of how well the corrugated board would perform in a box under load, such as it would experience at the bottom of a pallet. The findings are presented below.

- ECT performance was found to be strongly correlated with both washboarding depth and paper grammage. FEA was used to clarify these relationships. It was found that both the geometry of the washboarding due to changing the simulated glue force and the changing stiffness of the paper affected the resultant ECT performance. The relationships between these parameters were mapped and can be seen in the result section [6.4.1]. In general, an

increase in glue quantity decreased ECT performance while an increase in the elastic modulus of the paper (grammage) increased ECT performance.

- Three-point bend testing found that a doubling in washboarding depth increased three-point bend performance by only 3%. FEA confirmed this result.
- MD-Shear measurement showed an increase in performance for this test as washboarding depth increased. The increase in performance observed in physical testing (7 %) was roughly double the increase (4 %) seen in FEA when washboarding depth increased from 25 μm to 40 μm . This was possibly due to the differences in samples size, where the physical size of an MD-Shear test piece was larger than that modelled in FEA due to limitations in computer technology or as a result to the limitations of the isotropic elastic modulus used for FEA modeling.

The three measures of ECT, three-point bending and MD-Shear display a different effect on the performance of corrugated board when washboarding depth increases due to an increase in simulated glue force. The ECT showed a lowering of performance, three-point bending performance had significant change, and MD-Shear had an increase in performance. The effect of washboarding upon the performance is summarized in figure [7.1].





An increase in washboarding	Changes corrugated cardboard performance as follows		
	<i>ECT</i>	<i>Three-point bend</i>	<i>MD-Shear*</i>
			

Figure [7.1] – The effect of washboarding upon corrugated performance measures. *For washboarding depths less than 40 μm .

It must be noted though, using FEA modelling, that any gain in MD-Shear performance was lost for washboarding depths greater than 40 μm . Washboarding depths in excess of 40 μm are often achieved, especially on the inside of a box as this is not seen at the point of sale.

To what degree each performance measure relates to the final box strength is unclear, but these results show that the degree of washboarding depth may need to be taken into consideration when designing corrugated cardboard packaging for a specific application.

These studies were undertaken because little has been published about washboarding and its effect upon corrugated cardboard packaging. Many questions remain unanswered, which may be answered by more specific and in depth studies, where sections of this thesis may form the basis for further study on washboarding. A few examples of possible further studies are presented next.

Suggestions for further work

There are a number of potential areas for investigation that have not been covered with in this thesis. Some of these possible studies are presented below.

A study of the furnish of paper and its effect upon on washboarding for constant and cyclic humidity

The furnish of paper can vary in its content of recycled fibres and virgin fibres, which may be a mix of conifer fibres and/or hard wood fibres which have either been mechanically or chemically pulped. The effects of changing the paper furnish upon washboarding and hence corrugated cardboard boxes have had little investigation.

Virgin fibre is believed to perform better than recycled fibre in a humid environment, especially if the humidity is cycled. The washboarding depth of corrugated paper could be investigated for different paper furnishes for steady state moisture content (constant humidity and temperature) and/or dynamic moisture content (cyclic humidity).

A study into starch additives and its effect upon washboarding

It was shown in this thesis that the shrinkage of starch due to the reduction of moisture of the starch produces washboarding. A study could be performed by altering the properties of the starch. Some parameters that may be investigated include:

- Initial moisture content of the starch and its effect upon washboarding , and/or
- Additives to starch to reduce shrinkage or to weaken the starch (reduce the elastic modulus) and therefore reducing washboarding.

A study into washboarding and its effect upon box compression strength and survival

In this thesis the effect simulated glue force and therefore washboarding severity had upon the performance of three tests was investigated ie. three-point bend, ECT and MD-Shear. The

natural progression for this would be a study on how washboarding depth affects performance of the final box rather than the constituent cardboard. This could be done experimentally in an environment with constant humidity and temperature and modelled using FEA. This more extensive modelling was not attempted in this thesis due to computer limitations, but as computers get faster and increase in memory capability, this should become a viable investigation.

A study on washboarding variability

An investigation into how washboarding depth varies across the deckle could be studied. The deckle of the paper is the full width of the paper as it goes through the paper machine, which may be anything up to 6 metres wide. The properties of paper, such as strength, vary across the deckle, sometimes considerably (50 % variation) with the paper being generally stronger nearer the edge of the deckle.

The starch quantity applied may oscillate in the process of manufacturing corrugated board, which may be due to various resonances of the starch bath or corrugator surface tension. An investigation of the change of washboarding over time may help in process optimisation of the corrugator starch application.

A study into how printing quality could be improved by increasing the moisture content of the corrugated cardboard

It was shown in this thesis that the washboarding depth was reduced for levels of higher humidity. This may lead to improvements in the printing quality of corrugated cardboard by increasing the moisture content of the paper at the time of printing. This may be accomplished by either increasing the relative humidity of the environment or decreasing the temperature of the paper while keeping the relative humidity the same.

References

- [1] Wright, P.G.; McKinlay, P. R.; Shaw E. Y. N. 1992: Corrugated Fibreboard Boxes – Their design, use, quality control and testing, Fourth Edition. Amcor Fibre Packaging, Australia.
- [2] Ollett, A.L.; Parker, R.; Smith, A.C. 1991: Deformation and fracture behaviour of wheat starch plasticized with glucose and water. *Journal of Material Sciences*, Volume 26: 1351-1356
- [3] Netz, E. 1998: Washboarding and Print quality of corrugated board. *Packaging Technology and Science*, Volume 11: 145-167
- [4] Reich, M.; Allan, R. 1996: Measurement of washboarding of corrugated cardboard using digital image profilometry. In *Machine Vision Applications in Industrial Inspection IV -IS&T /SPIE Proceedings Volume 2665*
- [5] Gomez, K. 1994: Evaluation of a structured light technique in determining the surface profile of corrugated board. Honours thesis, Department of Applied Physics, Royal Melbourne Institute of Technology, Australia.
- [6] Nordstrand, T. 2003: Basic testing and strength design of corrugated boards and containers. Doctoral Thesis, Structural Mechanics, Lund University, Sweden
- [7] Cook, R.D. 1995: Finite element modelling for stress analysis. Wiley, New York USA.
- [8] Cook, R.D. 2001: Concepts and applications of finite element analysis, 4th edition. Wiley, New York USA.
- [9] Komzsis, L. 2005: What every engineer should know about computational techniques of finite element analysis. Taylor & Francis, Florida USA.
- [10] Baker, C.F.; Punton, V.W. 1989: Fundamentals of papermaking. Transactions of the ninth fundamental research symposium
- [11] Haslach, Jr. H.W. 2000: The moisture and rate-dependent mechanical properties of paper: A review. *Mechanics of Time-Dependent Materials*. Volume 4: 169-210
- [12] Page, D.H. 1985: The mechanism of strength development of dried pulps by beating. *Svensk Papperstidning*. Volume 88, No. 3: 30-35
- [13] Niskanen, K. 1998: Paper Physics. Published by Fapet Oy. in cooperation with the Finish Paper Engineers' Association and TAPPI.
- [14] Ruud,* A.; Bottergaard, O. 1991: The influence of the refining consistency on the physical properties of paper. TAPPI proceedings, 1991 International Paper Physics Conference: 463 – 472

- [15] Wistara, N.; Young, R.A. 1999: Properties and treatments of pulp from recycled paper: Part 1. Physical and chemical properties of pulps. *Cellulose*. Volume 6: 291-324
- [16] Stratton, R.A. 1991: Characterization of fiber-fiber bond strength from paper mechanical properties. TAPPI proceedings, 1991 International Paper Physics Conference: 561 – 577
- [17] Watson, A.; McKenzie, A.; Bland, D.; Dench, I.; Paterson, L.; Lampard, D.; Dawson, M.; Sloman, R.; Dobbie, M.; Martin, J. *Pulp & Papermaking*, notes for units 1 – 6. Victorian Tafe. Off-Campus Network.
- [18] Dasgupta, S. 1994: Mechanism of paper tensile-strength development due to pulp beating. *Tappi. Journal*. Volume 77, No. 6: 158-166
- [19] Bult, A.; Allan, R.; Porteus, N. 2000: Comparison of long term stacking performance of virgin and recycled liners- Report and Discussion Paper- Part 1. Project Report, AMCOR Research and Technology, Alphington, Australia.
- [20] Buzzi, V.; Allan, R.J. 1998: Project 50368R: Report for 1998 Part 1- Relative Performance of Virgin & Recycled Liners and Mediums. Project Report, AMCOR Research and Technology, Alphington, Australia.
- [21] Donega, L.H.; Eusufzai, A.R.K.; Mark, R.E. 1990: Compressive creep of virgin and recycled paperboard in steady and cyclic humidity environments. SUNY College of Environmental Science and Forestry, ESPRI Research Reports. Volume 93: 107 – 117
- [22] Fahey, D.J.; Bormett, D.W. 1982: Recycled fibres in corrugated fibreboard containers. *Tappi Journal*. Volume 65, No.10: 107-110
- [23] Benson, R.E. 1971: Effects of relative humidity and temperature on tensile stress-strain properties of kraft linerboard. TAPPI, Volume 54, No. 5: 699-703
- [24] Wink, W.A. 1961: The effect of relative humidity and temperature on paper properties. *Tappi Journal*. Volume. 44, No.6: 171A-178A
- [25] Skogman, R.T.; Scheie C.E. 1969: The effect of temperature on the moisture adsorption of Kraft paper. *Tappi Journal*. Volume 52, No. 3: 489-490
- [26] Salmén, N.L.; Back, E. L. 1977: *Svensk Papperstidning-Nordisk Cellulosa*. Volume 80, No. 6: 178
- [27] Zauscher, S.; Caulfield, D. F.; Nissan, A. H. 1996: The influence of water on the elastic modulus of paper. Part 1: Extension of H-bond theory. *TAPPI Journal*, December: 178-182.
- [28] Chatterjee, S.G.; Ramaro B.V.; Tien C. 1997: Water-vapor sorption equilibria of a bleached-kraft paperboard- A study of the hysteresis region. *Journal of Pulp and Paper Science*. Volume 23, No. 8: J366-J373

- [29] Ebrahimzadeh, P.R.; Kubát, J.; McQueen, D.H. 1996: Dynamic mechanical characterization of mechanosorptive effects in wood and paper. *Holz als Roh- und Werkstoff* Volume 54: 263-271
- [30] Australian Standard AS-1301.414 – Conditioning of paper for testing.
- [31] de Ruvo, A.; Lundberg, R.; Martin-Löf, S.; Söremark, Christer. 1976: The influence of temperate and humidity on the elastic and expansional properties of paper and the constituent fibre. *British Paper and Board Makers Association, Cambridge Symposium, 1973. Technical Section 785: 3-13*
- [32] Niskanen, K.J.; Kuskowski, S.J.; Bronkhorst, Curt A 1997: Dynamic hygroexpansion of paperboards. *Nordic Pulp and Paper Research Journal. Volume 12, No. 2: 102-109*
- [33] Forseth T; Helle T. 1997: Effect of moistening on cross-sectional details of calendered paper containing mechanical pulp. *Journal of Pulp and Paper Science. Volume 23, No. 3: 95-100*
- [34] Chalmers, I.R. 1996: The effect of humidity on packaging grade paper elastic modulus. *Appita: 779-785*
- [35] Soremark, C.; Fellers, C. 1991: Mechano-sorptive creep and hygroexpansion of corrugated board in bending. *1991 International Paper Physics Conference: 549-559*
- [36] Norman, B. 1989: Overview of the physics of forming. *Fundamentals of Papermaking (Baker, C.F.; Punton, V.W. Eds.). Mechanical. Engineering Publications Limited, London. Volume 3: 73-149*
- [37] Kinsky, R. 1993: *Engineering Mechanics and Strength of Materials. Mc-Graw-Hill Book Company, Sydney, Australia.*
- [38] Zauscher, S.; Caulfield, D.F.; Nissan, A.H. 1997: Influence of water on the elastic modulus of paper. Part 2: Verification of the H-bond theory. *TAPPI Journal. Volume 80, No. 1: 214-223*
- [39] Padanyi, Z.W. 1993: Physical Aging and Glass Transition Effects on the Mechanical Properties of Paper and Boxes. *Transactions of the Tenth Fundamental Research Symposium, Oxford. Volume 1: 521-545*
- [40] Panek, J.; Fellers, C.; Haraldsson, T. 2004: Principles of evaluation for the creep of paperboards in constant and cyclic humidity. *Nordic Pulp and Paper Research Journal. Volume 19, No. 2: 155-163*
- [41] Haslach, Jr. H.W. 1994: The Mechanics of Moisture Accelerated Creep in Paper. *TAPPI. Volume 77, No. 10: 179-186*
- [42] Alfthan, J.; Gudmundson, P.; Ostlund; S. 2002: A micromechanical model for mechanosorptive creep in paper. *Journal of Pulp and Paper Science. Volume 28, No. 3: 98-104*
- [43] Padanyi, Z.V. 1991: Mechano-sorptive effects and accelerated creep in paper. *TAPPI proceedings, 1991 International Paper Physics Conference: 397 – 411*

- [44] Habeger, C.C.; Coffin D.W. 2000: The role of stress concentrations in accelerated creep and sorption-induced physical aging. *Journal of Pulp and Paper Science*. Volume 26, No. 4: 145-157
- [45] Byrd, V. L. 1972: Effect of relative humidity changes on compressive creep response of paper. *TAPPI Journal*, Volume 55, No. 11: 1612 – 1613
- [46] Zhang G.; Niskanen K.; Tanaka A.; Keranen J.; Timofeev O. 2003: Effect of drying and fibre chemical composition on the creep strain of paper. *Journal of Pulp and Paper Science*. Volume 29, No. 7: 213-219
- [47] Loewen, S.R.; Foulger, M. 2002: TSO and fibre orientation: an introduction. *Pulp and Paper Canada*. Volume 103, No. 5: 42-46
- [48] Page, D.H. 1969: The structure and properties of paper: Part 1- The structure of paper. *Trend*. Volume 15: 7
- [49] Page, D.H. 1971: The structure and properties of paper: Part 2- Shrinkage, dimensional stability and stretch. *Trend*. Volume 18: 6-11
- [50] Loewen, S.R. 1997: Fibre orientation optimization. *Pulp and Paper Canada*. Volume 98, No. 10: 391-394
- [51] Serway, R.A. 1990: .Physics for scientists and engineers with modern physics, 3rd edition. Saunders College Publishing, USA: 301
- [52] Coffin, D.W.; Lif, J.O., Fellers; C. 2004: Tensile and ultrasonic stiffness of paper at different moistures- a clarification of the differences. *Nordic Pulp and Paper Research Journal*. Volume 19, No. 2: 257-263
- [53] Baum, G.A.; Habeger, C.C.; Fleischman, E.H. 1983: in *The Role of Fundamental Research in Paper Making* (Brander, J., Ed.). Mechanical Engineering Publications Limited, London. Volume 1: 453-478
- [54] Walter, J.B.; Telschow, K.L.; Gerhardstein, J.P.; Pufahl, B.M.; Habeger, C.C., Lafond; E.M.; Brodeur, P.H. 1999: Fabry-Perot laser ultrasonic elastic anisotropy measurements on a moving paper web. 26th Annual Review of Progress in Quantitative Nondestructive Evaluation, 1999.
- [55] Ridgway, P.L.; Russo, R.E.; Lafond, E.F.; Habeger, C.C.; Jackson, T. 2003: Laser ultrasonic system for on-line measurement of elastic properties of paper. *Journal of Pulp and Paper Science*. Volume 29, No. 9: 289-293
- [56] Batten, G.L. 1990: in *Material Interactions Relevant to the Pulp, Paper and Wood Industries*. Symposium Proceedings, Materials Research Society, Pittsburgh. Volume 197: 173-181
- [57] Blennow, A.; Bay-Smidt; A.M.; Leonhardt; P., Bandsholm, O.; Madsen, M.H. 2003: Starch paste stickiness is a relevant native starch selection criteria for wet-end paper manufacturing. *Starch/Starke*. Volume 55:381-389

- [58] Snyder, P.A. 1995: Controlling adhesive application. Flexo. September: 24-27
- [59] Ellis, R.P.; Cochrane, M.P.; Dale, M.F.B.; Duffus, C.M., Lynn, A., Morrison, I.M., Prentice, R.D.M., Swanston, J.S., Tiller, S.A. 1998: Starch production and industrial use. Journal of the Science of Food and Agriculture. Volume 77: 289-311
- [60] Zumdahl, S. 1998: Chemical Principles, 3rd Edition. Houghton Mifflin Company, Boston, USA.
- [61] Kirby, A.R.; Clark, S.A.; Parker, R.; Smith, A.C. 1993: The deformation and failure behaviour of wheat starch plasticized with water and polyols. Journal of Material Science, Volume 28: 5937-5942
- [62] Shen, M.C.; Eisenberg, A. 1966: Solid State Chemistry, Volume 3: 407
- [63] Willett, J.L.; Doane, W.M. 2002: Effect of moisture content on tensile properties of starch/poly(hydroxyester ether) composite materials. Polymer. Volume 43: 4413-4420
- [64] Greenspan, L. 1977: Humidity Fixed Points of Binary Saturated Aqueous Solutions. Journal of Research of the National Bureau of Standards-A. Physics and Chemistry, Volume 81A, No. 1, January-February
- [65] Stolpe, L. 2001: Corrugated board a basis for beauty. International Paperworld, Journal 9: 14-18
- [66] Cook, N. 2000: Label Non-Adhesion- Report No. 2. Test Report, AMCOR Research and Technology, Alphington, Australia.
- [67] Hallberg, E.; Glasenapp, A.O.; Lestelius, M. 2005: Quantification of banding on printed corrugated board using spatial frequency analysis. Packaging Technology and Science. Volume 18: 89-95
- [68] McGrattan, W. 1990: Key characteristics of linerboard, corrugating medium, and roll stock mechanical condition and their influence on the manufacture of corrugated products, part 1. TAPPI Journal, November: 99 –108
- [69] Zang, Y.H.; Aspler, J.S. 1995: Factors that affect the flexographic printability of linerboards. TAPPI Journal 78, No. 10: 240-23 – 240-33
- [70] Laschitz. S. 2002: Wellpappe: Einige gedanken zum waschbretteffekt und zu wellpappenkrümmungen- Einfluss des rohpapiers. Wochenblatt für Papierfabrikation. Volume 130, No. 8: 512-517
- [71] Collier, S. 2001: Advancing starch development to aid quality control on the corrugator. 2001 Developments in Manufacture, Technology and Markets for Corrugated Board, Manchester UK. Pira International, Paper 14
- [72] Coleman, M. 2001: Box printing in a multimedia world. Boxboard Containers International, January: 20-24
- [73] Kamp, F. 2001: Corrugated board and pre-printed liner update. Flexo Tech. October/November: 22-24

- [74] Haglund, N. 1998: Improving the printability of kraftliner. Paper Technology and Industry, August: 265-267
- [75] Arimond, J.; Koss, M.S. 1999: Mechanical characteristics. Asia Pacific Paperboard Packaging, January: 32-34
- [76] Selway, J.W.; Kirkpatrick, J. 1992: The assessment of high humidity corrugated box performance. A Discussion of Theoretical and Experimental Factors. TAPPI Cyclic Creep Symposium, September.
- [78] Carabin, P.; Kerr, R.B. 1994: Visual Ranking of Newsprint Colour by the Rangefinder Method. Journal of Pulp and Paper Science, Volume 20 No. 2: J66 – J69
- [79] Lyne, M.B. 1979: Multidimensional scaling of print quality. TAPPI Journal, Volume 62, No 11: 103-107
- [80] Pommier and Poustis J. 1991: Bending Stiffness of corrugated Board Prediction Using the Finite Element Method . ASME, Volume 112, editor R.W.Perkins: 67-70.
- [81] Boas, M.L. 1983: Mathematical methods in the physical sciences, 2nd edition. John Wiley & Sons, Singapore: 297-335
- [82] Lynn, P.A.; Fuerst, W. 1989: Digital Signal Processing with Computer Applications. John Wiley & Sons, Inc.
- [83] Oppenheim, A.V.; Schafer, R.W.; Buck, J.R. 1999: Discrete-time signal processing. Prentice Hall, New Jersey USA.
- [84] Bentley, J.P. 1983: Principles of Measurement Systems, Second Edition. John Wiley & Sons, Inc.
- [85] Van Grieken, R.E.; Markowicz, A. A. 2002: Handbook of X-Ray Spectrometry, 2nd edition, Volume 29. Marcel Dekker Inc., New York, USA.
- [86] Nomura Shoji- Sonic Sheet Tester. http://www.nomurashoji.com/e_sst.html
- [87] Australian Standard AS 1301.444s-1992 - Edgewise compression resistance of corrugated fibreboard
- [88] MSC.Mentat preprocessor and MSC.Marc Finite Element Analysis solver. <http://www.mscsoftware.com>
- [89] Fellers, C. 1980: The significance of structure on the compression behaviour of paper. Doctoral dissertation, Royal Institute of Technology, Stockholm, Sweden.

Appendix - A

Washboard profiling program

This appendix includes the main source code for the program that measures washboarding profiles. Figure [A.1] shows a screen capture of the main window of the program that has a image captured and the washboarding profile calculated and shown on the image. The red lines on the image are the profiles that have been phase unwrapped but not filtered. The yellow washboarding profile is the average of the filtered red profiles. The details of the techniques used by the program to extract the final profile (yellow line) are in method section [4.1].

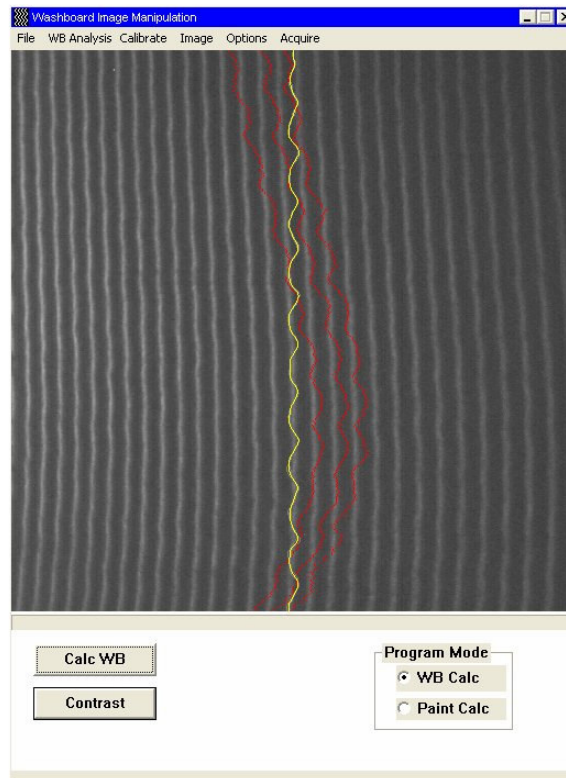


Figure [A.1] – Screen capture of main window of washboarding profiling program showing a captured image and the extracted non-filtered profiles (red lines) and the final washboarding profile (yellow line) for the image.

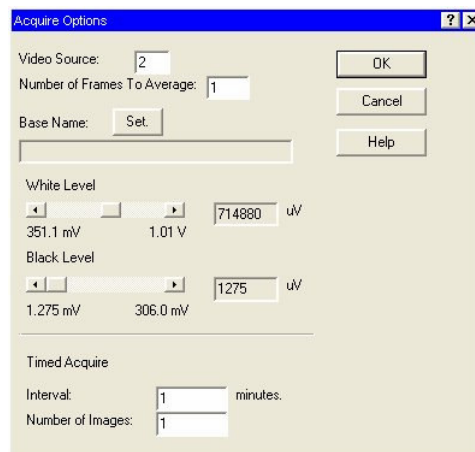


Figure [A.2]- Screen capture of acquisition options.

Figure [A.2] shows the various image capturing options Figure [A.3] shows filtering options available for the filtering of the final profile and figure [A.4] shows calculation options, including the setting for calibrations, ie. the image width as measured from a captured 1 mm ruled grid; the width of the calibration board; and the height of the board at that width (used for calculating the angle). See method section [] for details of calibration.

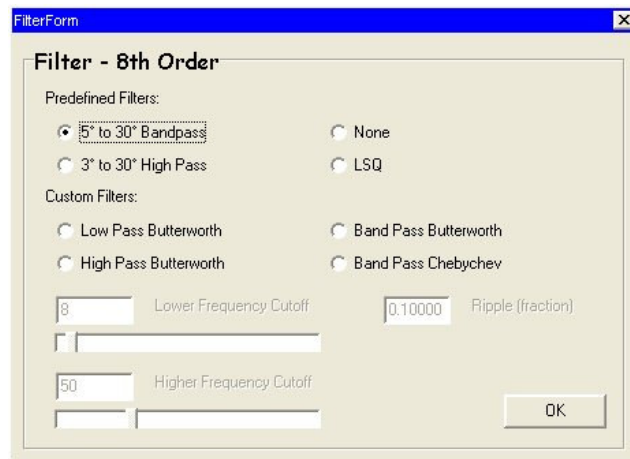


Figure [A.3]- Screen capture of filtering options

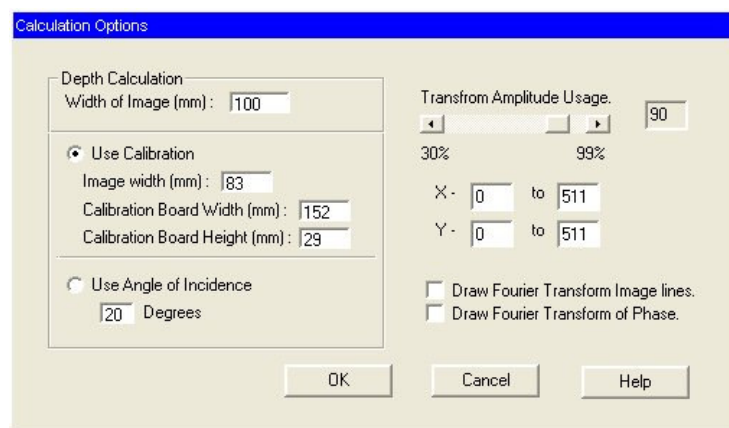


Figure [A.4] - Screen capture of calculation options, including calibration settings.

Following, is the source code and header files used for the washboarding profiling program, which uses techniques covered in section [1.] in both the theory and methods chapters. SingFFT32.c was written by R. C. Singleton in Fortran and translated to “c” by Javier Soley.

Image67forbuilder.h
Main header file for the program

```
#define CM_EXIT _OWL_CM_EXIT
#if !defined(IMAGE_H)
#define IMAGE_H

#define _USE_OWL_CM_EXIT

#include <pch.h>
#include <gdiobjec.h>
#include "imageacqanl.rh"
#include <string.h>
#include <decframe.h>
#include "Pictwind.h"
#include <button.h>
#include "dialogs60.h"
#include <gdiobjec.h>
#include <opensave.h>
#include <gauge.h>
#define _USE_OWL_CM_EXIT
#define _OWLVCLPCH
#include <owl/pch.h>
#include <math.h>
#include <inputdia.h>
#include <applicat.h>
#include <framewin.h>
#include <dialog.h>
#include <statusba.h>
#include <dc.h>
#include <stdlib.h>
#include <string.h>
#include <stdio.h>
#include <checkboxox.h>
#include <edit.h>
#include <button.h>
#include "olwintyp.h"
#include "olimgapi.h"
#include "olfgapi.h"
#include "unit1.h"
#include "shape.h"

unsigned char* ImageP;
TFilterForm* dlg;

typedef unsigned char uint8;
const int IDC_LEDGAUGE = 209;
const uint16 ID_BUTTON = 1000;
const uint16 ID_SETA = 1001;
const uint16 ID_SETB = 1002;
const uint16 ID_WB = 1003;
const uint16 ID_PAINTRATIO = 1004;
const uint16 ID_WBGROUP = 1005;
const uint16 ID_CALCPR = 1006;
const uint16 ID_PROUTPUT = 1007;
const uint16 ID_CONTRAST = 1008;

/* Acquire Variables*/
extern OLT_APISTATUS Status;
extern int iCount;
extern OLT_IMGDEVINFO DevInfoList;
extern char* Alias;
extern OLT_IMG_DEV_ID DeviceId;
extern OLT_FG_FRAME_ID FrameId;
extern OLT_FG_FRAME_INFO FrameInfo;
extern USHRT NewSource,OldSource;
extern bool AcqSet;

extern LoadBuffer(int Width,int Height,uint8* cBuffer);
extern int SetLevels(long WhiteLevel, long BlackLevel);
extern int SetAcqDev(void);
extern void KillAcqDev(void);
extern "C" fft(float*,float*,long,long,long,int);
extern void InitOptionsDialog(void);
extern void InitAcquireOptionsDialog(void);
extern struct TDialoglTransferBuffer DialoglBuffer;
extern struct TOptionsTransferBuffer OptionsBuffer;
extern struct TAcquireOptionsBuffer AcquireOptions;
extern struct TInfoBuffer InfoBuffer;
extern struct TCalibrateBuffer CalibrateBuffer;

namespace OWL {

class TDrawApp : public TApplication {
public:
    TDrawApp() : TApplication() {}
    void InitMainWindow();
};

class TFFTWindow : public TFrameWindow {
public:
    TFFTWindow(TWindow* , const char far* ,TWindow* , bool shrink );
    ~TFFTWindow();
    void DrawFFTlines(ClassLib:TRect&);
    void LoadArray(float array[512][256],long lengthX,int numlines );
    void Paint(IDC&, bool, ClassLib:TRect&);

protected:
    float FFTlinesArray[512][256];
    long FFTlength;
    int FFTnumlines;
};
};
```

Image67forbuilder.h
Main header file for the program

```
class TDrawWindow : public TWindow {
public:
    int numofcolours;
    char NameOfFile[100];

    TDrawWindow(TWindow* parent = 0);
    ~TDrawWindow();

    LOGPALETTE* MyLogPalette;

    TFFTWindow* tmpwin;
    float* FFTreal;
    float* FFTimg;
    float* Phase;

    uint16* Prime;

    uint8* Image;
    uint8* ScreenImage;
    RGBQUAD* mono;
    ClassLib::TPoint StartSelection;
    ClassLib::TPoint EndSelection;
    ClassLib::TPoint oldpoint;
    ClassLib::TPoint oldpointA;
    ClassLib::TPoint oldpointB;

    TFont* SmallFont;
    ClassLib::TPoint SetA_SPoint;
    ClassLib::TPoint SetA_EPoint;
    ClassLib::TPoint SetB_SPoint;
    ClassLib::TPoint SetB_EPoint;
    TButton* SetAButton;
    TButton* SetBButton;
    TGroupBox* WBGroup;
    TButton* CalcWB;
    TButton* CalcPR;
    TButton* Contrast;
    TEdit* PROutput;

    long length;
    float Cf; // conversion factor
    float* realPhase;
    float* finalPhase;
    float* BasePhase;
    float* BoardPhase;

    bool StartPal;
    bool HaveSelection, NewASelection, NewBSelection;
    bool NewSelection;
    bool ButtonDown;
    bool single;
    bool NewImage;

    bool seta;
    bool setb;
    bool WBcreated;
    bool PRcreated;
    TResultsDialog* RDialog;
    bool Timer1;

    TPointer<TFormShape> ShapeDlg;

protected:
    void filterarray(float *x, int size);
    void CmForm(void);
    void CmShape(void);
    void CmShape(void);
    TClientDC* VDC;
    // Override member function of TWindow
    bool CanClose();
    TGauge* LedGauge;
    // Message response functions
    void ThreshHold(void);
    void ThreshHoldArea(int x1, int y1, int x2, int y2, int minimum);
    void InitButtons(void);
    void CreatePR(void);
    void CreateWB(void);
    void KillPR(void);
    void KillWB(void);
    void CmSetWB(void);
    void CmSetPR(void);
    void CmCalcPR(void);

    void EvLButtonDown(uint, ClassLib::TPoint&);
    void EvLButtonUp(uint, ClassLib::TPoint&);
    void EvMouseMove(uint, ClassLib::TPoint&);
    //void EvPaint(void);
    void DoIt(void);
    void CmFileOpen();
    void LoadLogo();
    void CmOpenMult();
    void CmAutoCalc();
    void CmConvolve();
    void CmOptions();
    void CmTimedAcquire();
    void CmKillTimedAcquire();
    void EvTimer(uint);
    void CmAcquireOptions();
    void CmAcquire();
    bool SaveMultImage(void);
    void CmPreview();
    void CmFileCalibrate();
    void Paint(TDC&, bool, ClassLib::TRect&);
```

Image67forbuilder.h
Main header file for the program

```
void DrawBitmap(void);
void InitFFTreals(uint16,uint16);
void DrawSpectrum(uint16,float*);
void SmoothPhase(uint16);
void SmoothArray(float* ,uint16&);
float FindMaxNum(float* ,int);
float FindMinNum(float* ,int);
int WhereIsMaxNum(float* ,int);
int WhereIsMinNum(float* ,int);
float calc_phase(float a,float b);
int FindDominantFreq(int* , float*);
void PolyriseArray(float*,int);
void CmLSQ (void);
void AdjustLength(int &);
    int CalcAndDrawPhase(int length,int majfreq);
float getpeaks(float* ,int, float& );
void GiveResults(float* , int,float*);
void CheckXLength(void);
int GetPhaseOffline(int ,float*);
void UpdateGauges(uint);
    void SetupWindow();
    void InitScreenImage();
    void CmSetA(void);
    void CmSetB(void);

    void CmExtractBase(void);
    void CmExtractBoard(void);
    void CmCalibrate(void);
    void CmCurrentCf(void);
    void CmSetCf(void);
    float Slope(float* y);

    DECLARE_RESPONSE_TABLE(TDrawWindow);
};

} //OWL
#endif
```

Image.cpp
Main source code for extraction of washboarding profiles

```
#define _USE_OWL_CM_EXIT
#define _OWLCLPCH
#include <owl/pch.h>
#include <classlib/pointer.h>
#include "adopt.h"
#include "Image67forbuilder.h"
#include "imageview4bb.h"
#include <applicat.h>
#include <framewin.h>
#include <dc.h>
#pragma hdrstop
#include "filterpoles.h"
#include "lsqpoly.h"
#include "unit1.h"
#define ERZERO 1e-5

DEFINE_RESPONSE_TABLE1(TDrawWindow, TWindow)
    EV_WM_TIMER,
    EV_WM_PAINT,
    EV_WM_LBUTTONDOWN,
    EV_WM_LBUTTONUP,
    EV_WM_MOUSEMOVE,
    EV_WM_RBUTTONDOWN,
    EV_COMMAND(CM_FORM, CmForm),
    EV_COMMAND(CM_SHAPE, CmShape),
    EV_COMMAND(102, CmFileOpen),
    EV_COMMAND(CM_OPENMULT, CmOpenMult),
    EV_COMMAND(CM_AUTOOPEN, CmAutoCalc),
    EV_COMMAND(CM_CONVOLVE, CmConvolve),
    EV_COMMAND(902, CmLSQ),
    EV_COMMAND(CM_OPTIONS, CmOptions),
    EV_COMMAND(CM_ACQUIREOPTIONS, CmAcquireOptions),
    EV_COMMAND(CM_ACQUIRE, CmAcquire),
    EV_COMMAND(CM_TIMEDACQUIRE, CmTimedAcquire),
    EV_COMMAND(CM_KILLTIMEDACQUIRE, CmKillTimedAcquire),
    EV_COMMAND(CM_PREVIEW, CmPreview),
    // EV_COMMAND(CM_FILECALIBRATE, CmFileCalibrate),
    EV_COMMAND(CM_EXTRACTBASE, CmExtractBase),
    EV_COMMAND(CM_EXTRACTBOARD, CmExtractBoard),
    EV_COMMAND(CM_CALCULATE, CmCalibrate),
    EV_COMMAND(CM_CURRENTCF, CmCurrentCf),
    EV_COMMAND(CM_SETCF, CmSetCf),
    EV_COMMAND(ID_BUTTON, DoIt),
    EV_COMMAND(ID_CALCPR, CmCalcPR),
    EV_COMMAND(ID_SETA, CmSetA),
    EV_COMMAND(ID_SETB, CmSetB),
    EV_COMMAND(ID_WB, CmSetWB),
    EV_COMMAND(ID_PAINTRATIO, CmSetPR),
    EV_COMMAND(ID_CONTRAST, ThreshHold),
END_RESPONSE_TABLE;

void TDrawWindow::CmForm(void)
{
    dlg -> ShowModal();
}
void TDrawWindow::CmShape(void)
{
    ShapeDlg -> Show();
    ShapeDlg -> DoExtraction();
}
void TDrawWindow::CmExtractBase(void)
{
    for (int i = 0 ; i < 512; i++){
        BasePhase[i]=finalPhase[i];
    }
}
void TDrawWindow::CmExtractBoard(void)
{
    for (int i = 0 ; i < 512; i++){
        BoardPhase[i]=finalPhase[i];
    }
}

void TDrawWindow::CmCalibrate(void)
{
    float PhaseBaseSlope, PhaseBoardSlope, PhaseSlope;

    PhaseBaseSlope = Slope(BasePhase);
    PhaseBoardSlope = Slope(BoardPhase);
    PhaseSlope = -PhaseBaseSlope + PhaseBoardSlope;

    float BoardSlope, Iw, Bw, Bh;
    Iw = atof(OptionsBuffer.cIw);
    Bw = atof(OptionsBuffer.cBw);
    Bh = atof(OptionsBuffer.cBh);
    BoardSlope = Bh/sqrt((Bw*Bw)-(Bh*Bh));

    Cf = (BoardSlope*Iw)/(PhaseSlope*512.0) *1000; // 1000 for um/rad not mm/rad
    CmCurrentCf();
}
void TDrawWindow::CmSetCf(void)
{
    char buffer[20];
    buffer[0] = NULL;
    TInputDialog(this, "Set Conversion Factor (um/rad)", "Input New Conversion Factor", buffer, sizeof(buffer)).Execute();
    Cf= atof(buffer);
    CmCurrentCf();
}

void TDrawWindow::CmCurrentCf(void)
```

Image.cpp
Main source code for extraction of washboarding profiles

```
{
    char tmp[20];
    sprintf(tmp,"%%.4g um/rad", Cf);
    MessageBox(tmp,"Conversion Factor.",MB_OK);
}
float TDrawWindow::Slope(float* y)
{
    float SumX=2, SumY=2, SumXY=2, SumX_2=2;
    float a;
    int n = 512;
    for(int x = 0; x< n; x++){
        SumX += float(x);
        SumY += /*x*/y[x];
        SumXY += float(x)*/*x*/y[x];
        SumX_2 += float(x)*float(x);
    }

    a = (float(n)*SumXY - (SumX*SumY))/((float(n)*SumX_2)-(SumX*SumX));
    return (a);
}

bool
TDrawWindow::SaveMultImage(void)
{
    FILE* out;
    char file[120];
    char cFileNum[5];
    strcpy(file,AcquireOptions.MultFilename);
    strcat(file,itoa(AcquireOptions.FileNum++, cFileNum,10));
    strcat(file, ".img");

    if ((out = fopen(file,"wb"))==NULL){
        return false;
    }
    fwrite(Image,1,512*512,out);
    fclose(out);
    return true;
}

void
TDrawWindow::CmTimedAcquire()
{
    if(AcquireOptions.FileNameSet == false){
        MessageBox("Set Base Name under Options|Acquire","File Name Base Not Set.", MB_OK);
        return;
    }
    if(AcquireOptions.IntervalSet == false){
        MessageBox("Set Interval under Options|Acquire","Interval Time Not Set.", MB_OK);
        return;
    }
    if(AcquireOptions.TimedImagesSet == false){
        MessageBox("Set Number under Options|Acquire","Number of Images Not Set.", MB_OK);
        return;
    }
    if (Timer1 == false){
        SetTimer(1,AcquireOptions.Interval*60L*1000L,0);
        Timer1 = true;
        CmAcquire(); // do an acquire at start of timing
    }
}

void
TDrawWindow::CmKillTimedAcquire()
{
    if (Timer1 == true){
        KillTimer(1);
        Timer1 = false;
    }
}

void
TDrawWindow::EvTimer(uint)
{
    if( AcquireOptions.TimedImages == 0){
        KillTimer(1);
        return;
    }
    CmAcquire();
    AcquireOptions.TimedImages-- ;
}

void
TDrawWindow::CmAcquire()
{
    if(AcquireOptions.FileNameSet == false){
        MessageBox("Set Base Name under Options|Acquire","File Name Base Not Set.", MB_OK);
        return;
    }
}

uint16* wholeimg;
wholeimg = (uint16*) new uint16 [512*512];

char message[30];
int errornum= 0, NumOfFrames = atoi(AcquireOptions.NumOfFrames);
NewImage = true;

if(PRcreated==true) KillPR();
if(WBcreated==true) KillWB();
SetA_SPoint =SetA_EPoint = oldpointA;
SetB_SPoint =SetB_EPoint = oldpointB;
```


Image.cpp
Main source code for extraction of washboarding profiles

```
errornum = LoadBuffer(512,512,Image);
for(int i =0 ; i< 262144; i++){
    wholeimg[i] = Image[i];        // initialise array
}
for(int f =1; (f < NumOfFrames)&&(errornum==0); f++){ // reduce noise
    errornum = LoadBuffer(512,512,Image);
    for(int i =0 ; i< 262144; i++){
        wholeimg[i] += Image[i];
    }
}
single = true;

if (errornum ==0){
    for(int i =0 ; i< 262144; i++){
        Image[i] = wholeimg[i]/NumOfFrames;        // reduce dynamic range to 0 to 256
    }
    DrawBitmap();
    if(SaveMultImage()== false){
        MessageBox("Could not write Image To disk","Disk Write Error", MB_OK);
        return;
    }
    HaveSelection = false;
    NewSelection = true;
    StartSelection.x =atoi(OptionsBuffer.X1);
    StartSelection.y =atoi(OptionsBuffer.Y1);
    EndSelection.x =atoi(OptionsBuffer.X2);
    EndSelection.y =atoi(OptionsBuffer.Y2);
}
else {
    sprintf(message,"Returned: %d",errornum);
    MessageBox(message,"Error Acquiring Image", MB_OK);
}

delete[] wholeimg;
}

void
TDrawWindow::CmPreview()
{
    uint16* wholeimg;
    wholeimg = (uint16*) new uint16 [512*512];

    char message[30];
    int errornum= 0, NumOfFrames = atoi(AcquireOptions.NumOfFrames);
    NewImage = true;

    if(PRcreated==true) KillPR();
    if(WBcreated==true) KillWB();
    SetA_SPoint =SetA_EPoint = oldpointA;
    SetB_SPoint =SetB_EPoint = oldpointB;

    errornum = LoadBuffer(512,512,Image);
    for(int i =0 ; i< 262144; i++){
        wholeimg[i] = Image[i];        // initialise array
    }
    for(int f =1; (f < NumOfFrames)&&(errornum==0); f++){ // reduce noise
        errornum = LoadBuffer(512,512,Image);
        for(int i =0 ; i< 262144; i++){
            wholeimg[i] += Image[i];
        }
    }
    single = true;

    if (errornum ==0){
        for(int i =0 ; i< 262144; i++){
            Image[i] = wholeimg[i]/NumOfFrames;        // reduce dynamic range to 0 to 256
        }
        DrawBitmap();

        HaveSelection = false;
        NewSelection = true;
        StartSelection.x =atoi(OptionsBuffer.X1);
        StartSelection.y =atoi(OptionsBuffer.Y1);
        EndSelection.x =atoi(OptionsBuffer.X2);
        EndSelection.y =atoi(OptionsBuffer.Y2);
    }
    else {
        sprintf(message,"Returned: %d",errornum);
        MessageBox(message,"Error Acquireing Image", MB_OK);
    }

    delete[] wholeimg;
}

void
TDrawWindow::ThreshHold(void)
{
    long i;
    int max, min;//, middle;

    max = 0;
    for(i= 0; i< (511L*512L); i++){
        if (Image[i] > max) max = Image[i];
    }

    min = max;
    for(i= 1; i< (511L*512L); i++){
        if (Image[i] < min) min = Image[i];
    }

    for(i= 1; i< (512L*512L); i++){
```

Image.cpp
Main source code for extraction of washboarding profiles

```

//      if (Image[i] < middle){
//          Image[i] = (Image[i]-min)*(255.0/(max-min));
//          ScreenImage[i]= 16 + (Image[i]* 239/255.0);
//          if( ScreenImage[i] <16) ScreenImage[i] = 32;
//      }
//      Invalidate(false);
//  }

void
TDrawWindow::ThreshHoldArea(int x1, int y1, int x2, int y2, int board)
{
    int x,y;
    int max, min;//, middle;

    max = 0;
    for(x= x1; x< x2; x++){
        for(y= y1; y< y2; y++){
            if (Image[x+(512*y)] > max) max = Image[x+(512*y)];
        }
    }
    min = 255;
    for(x= x1; x< x2; x++){
        for(y= y1; y< y2; y++){
            if (Image[x+(512*y)] < min) min = Image[x+(512*y)];
        }
    }

    for(x= x1; x< x2; x++){
        for(y= y1; y< y2; y++){
            /*      if (Image[x+(512*y)]>board){
            Image[x+(512*y)] = 255;
            }else{
            Image[x+(512*y)] = 0;
            } */

            float tmp= (float(Image[x+(512*y)])-float(min))*(255.0/float(board-min));
            if (tmp > 255) tmp = 255;
            Image[x+(512*y)] =tmp;

            ScreenImage[x+(512*y)] = 16 + ((Image[x+(512*y)]* 239)/256);
            if( ScreenImage[x+(512*y)] <16) ScreenImage[x+(512*y)] = 17;
        }
    }

    Invalidate(false);
}

void
TDrawWindow::CmCalcPR(void)
{
    float sum,sumD, AUI,/* PRatio,*/ dlDiff ;
    long x,y;
    char tmp[50];
    int i=0;

    for (sum = 0,i=0, y=SetB_SPoint.y; y <SetB_EPoint.y; y++){
        for (x=SetB_SPoint.x; x <SetB_EPoint.x; x++,i++){
            sum += Image[x+512L*y];
        }
    }
    for (sum = 0,i=0, y=SetD_SPoint.y; y <SetD_EPoint.y; y++){
        for (x=SetD_SPoint.x; x <SetD_EPoint.x; x++,i++){
            sumD += Image[x+512L*y];
        }
    }

    ThreshHoldArea(SetA_SPoint.x,SetA_SPoint.y,SetA_EPoint.x,SetA_EPoint.y, (int) (sum+sumD)/2);
    float cutoff;
    for (sum = 0,i =0, y=SetA_SPoint.y; y <SetA_EPoint.y; y++){
        for (x=SetA_SPoint.x; x <SetA_EPoint.x; x++,i++){
            if ( Image[x+512L*y] < 127){cutoff = 0;} else {cutoff = 255;}
            sum += cutoff;
        }
    }
    AUI = 1-(sum/(i*255.0));//(((SetA_EPoint.x-SetA_SPoint.x)*(SetA_EPoint.y-SetA_SPoint.y));

    //-----stddev-----
    float mean=0, Sx2=0, SD;
    for (sum = 0, x=SetA_SPoint.x; x <SetA_EPoint.x; x++){
        for (y=SetA_SPoint.y; y <SetA_EPoint.y; y++,i++){
            Sx2 += Image[x+512L*y]*Image[x+512L*y];
            mean += Image[x+512L*y];
        }
    }
    mean /= (float)i;
    SD = sqrt(Sx2/i - mean*mean);

    //-----end stddev-----
    // dlDiff = (Darkness - Lightness)/(Darkness+Lightness);

    sprintf(tmp,"Print Coverage = %.3f%%", AUI*100);
    PROutput->Clear();

```

Image.cpp
Main source code for extraction of washboarding profiles

```
    PROutput->Insert(tmp);
}

void
TDrawWindow::CmSetWB(void)
{
    if(PRcreated==true) KillPR();
    CreateWB();
}

void
TDrawWindow::CmSetPR(void)
{
    if(WBcreated==true) KillWB();
    CreatePR();
    // Threshold();
}

void
TDrawWindow::CreatePR(void)
{
    SetAButton -> Create();
    SetAButton -> SetWindowFont(*SmallFont,true);
    SetBButton -> Create();
    SetBButton -> SetWindowFont(*SmallFont,true);
    CalcPR -> Create();
    PROutput -> Create();

    seta = false;
    setb = false;
    PRcreated = true;
}

void
TDrawWindow::KillPR(void)
{
    SetAButton -> Destroy();
    SetBButton -> Destroy();
    CalcPR -> Destroy();
    PROutput -> Destroy();
    PRcreated = false;
}

void
TDrawWindow::CreateWB(void)
{
    CalcWB->Create();
    Contrast-> Create();
    WBcreated= true;
}

void
TDrawWindow::KillWB(void)
{
    CalcWB -> Destroy();
    Contrast -> Destroy();
    WBcreated = false;
}

void
TDrawWindow::CmSetA(void)
{
    seta = true;
    setb = false;
}

void
TDrawWindow::CmSetB(void)
{
    seta = false;
    setb = true;
}

TDrawWindow::TDrawWindow(TWindow* parent)
{
    Init(parent, 0, 0);

    SmallFont = new TFont("Times New Roman", 20);
    SetAButton = new TButton( this, ID_SETA, "Area U. I.", 20, 540, 112, 30);
    SetBButton = new TButton( this, ID_SETB, "Board Intsty.", 152, 540, 112, 30);
    CalcPR = new TButton( this, ID_CALCPR, "Calc SP", 20, 580, 112, 30);
    CalcWB = new TButton( this, ID_BUTTON, "Calc WB", 20, 540, 112, 30);
    Contrast = new TButton( this, ID_CONTRAST, "Contrast", 20, 580, 112, 30,true);
    PROutput = new TEdit( this , ID_PROUTPUT, "",20, 620,244,20);

    SetAButton -> DisableAutoCreate();
    SetBButton -> DisableAutoCreate();
    CalcWB -> DisableAutoCreate();
    CalcPR -> DisableAutoCreate();
    PROutput -> DisableAutoCreate();
    Contrast -> DisableAutoCreate();

    numofcolours = 254;

    WBGroup = new TGroupBox(this, ID_WBGROUP,"Program Mode", 330, 540, 127, 80);
    new TRadioButton( this, ID_WB, "WB Calc", 350, 560, 100, 24);
    new TRadioButton( this, ID_PAINTRATIO, "Paint Calc", 350, 590, 100, 20);

    WBcreated = false;
}
```

Image.cpp
Main source code for extraction of washboarding profiles

```
PRcreated = false;

LedGauge = new TGauge(this, IDC_LEDGAUGE, 0, 514, 512, 15 /*24*/);
LedGauge->SetRange(0, 100);
LedGauge->SetLed(4, 80);
LedGauge->SetStep(8);

RDialog= new TResultsDialog(this, DIALOG_1);
InitOptionsDialog();

AcqSet = false;
Timer1 = false;
AcquireOptions.IntervalSet = false;
AcquireOptions.TimedImagesSet = false;
AcquireOptions.Interval = 1;
AcquireOptions.TimedImages = 1;
AcquireOptions.FileNameSet = false;
AcquireOptions.WhiteLevel = 714880;
AcquireOptions.BlackLevel = 1275;
SetLevels(AcquireOptions.WhiteLevel, AcquireOptions.BlackLevel);
InitAcquireOptionsDialog();

realPhase =(float*) new float [512];
finalPhase =(float*) new float [512];
BasePhase =(float*) new float [512];
BoardPhase =(float*) new float [512];

Cf = 1; // cnver factor set to one so phase can still be found

Phase = (float*) new float [512];
FFTreal = (float*) new float [512];
FFTimg = (float*) new float [512];
Image = (uint8*) new uint8 [512*512];
ImageP = Image;
ScreenImage = (uint8*) new uint8 [512*512];
mono = (RGBQUAD*) new RGBQUAD[ 256];
HaveSelection = false;
ButtonDown = false;
NewSelection=NewASelection=NewBSelection= true;
StartPal = true;
NewImage = true;

MyLogPalette = (LOGPALETTE*)new char[sizeof(LOGPALETTE) + sizeof(PALETTEENTRY) * numofcolours];
MyLogPalette->palVersion = 0x300;
MyLogPalette->palNumEntries = (WORD) numofcolours;

for (int i = 0; i < numofcolours; ++i) {
    MyLogPalette->palPalEntry[i].peRed = mono[i].rgbBlue = uint8(i);
    MyLogPalette->palPalEntry[i].peGreen = mono[i].rgbGreen = uint8(i);
    MyLogPalette->palPalEntry[i].peBlue = mono[i].rgbRed = uint8(i);
    MyLogPalette->palPalEntry[i].peFlags = PC_RESERVED;
}

}

void
TDrawWindow::SetupWindow()
{
    AdoptVCLAppWindow(GetHandle());
    dlg = new TFilterForm(::Application);
    ShapeDlg = new TFormShape(::Application);
    TWindow::SetupWindow();
    UpdateGauges(0);
    LoadLogo();
    InitScreenImage();
    InitButtons();
}

void
TDrawWindow::UpdateGauges(uint setting)
{
    LedGauge->SetValue(setting);
}

TDrawWindow::~TDrawWindow()
{
    delete[] MyLogPalette;
    delete[] Image;
    delete[] FFTreal;
    delete[] FFTimg;
    delete[] Prime;
    delete[] ScreenImage;

    delete[] realPhase;
    delete[] finalPhase;
    delete[] BasePhase;
    delete[] BoardPhase;

    if (AcqSet == true) KillAcqDev();
    if (Timer1 == true) KillTimer(1);
}

void
TDrawWindow::CmLSQ(void)
{
    TImageView Cinema(this, CINEMA);
    // Cinema.Create();
    Cinema.PlayImages();
}
```

Image.cpp
Main source code for extraction of washboarding profiles

```

}

void TDrawWindow::filterarray(float *x, int size)
{
    float y[512], a, b, r, theta, PolesOnly[20][2];

    #define ORDER 8

    int whichfilter =1, zerotype;
    bool None =false;
    float Gain;

    switch (zerotype){
        case 1: a = -2; b = 1; break;
        case 2: a = 2 ; b = 1; break;
        case 3: a = 2 ; b = -1;
    }

    switch(dlg->Filter){

        case dlg->LSQ:
            whichfilter = 0;
            break;
        case dlg->None:
            None = true;
            break;
        case dlg->Normal:
            whichfilter = 1;
            FilterPoles(1,3,ORDER,5.0,30.0,NULL, PolesOnly,&zerotype, &Gain);
            break;
        case dlg->PreDefined2:
            whichfilter = 1;
            FilterPoles(1,3,ORDER,3.0,30.0,NULL, PolesOnly,&zerotype, &Gain);
            break;
        case dlg->LowPass:
            whichfilter = 1;
            FilterPoles(1,1,ORDER,dlg->FreqLow,NULL,NULL, PolesOnly,&zerotype, &Gain);
            break;
        case dlg->HighPass:
            whichfilter = 1;
            FilterPoles(1,2,ORDER,dlg->FreqLow,NULL,NULL, PolesOnly,&zerotype, &Gain);
            break;
        case dlg->Butt:
            whichfilter = 1;
            FilterPoles(1,3,ORDER,dlg->FreqLow,dlg->FreqHigh,NULL, PolesOnly,&zerotype, &Gain);
            break;
        case dlg->Cheb:
            whichfilter = 1;
            FilterPoles(2,3,ORDER,dlg->FreqLow,dlg->FreqHigh,dlg->Ripple, PolesOnly,&zerotype, &Gain);
            break;
    }

    switch (zerotype){
        case 1: a = 2; b = 1; break;
        case 2: a = -2 ; b = 1; break;
        case 3: a = 0 ; b = -1;
    }

    if(None == false){
        if(whichfilter){ // if a filter is picked
            for(int n = 0 ; n < size; n++){
                x[n] /= Gain; // brin gain to 0db // rem for biggy
            }
            for(int i = 0 ; i < (ORDER /2); i++){
                y[0] = 0;
                y[1] = 0;
                for(int n = 2 ; n < size; n++){
                    r = PolesOnly[i][0]; theta = PolesOnly[i][1];
                    y[n] = 2*r*cos(theta)*y[n-1] - r*r*y[n-2] + x[n] + a* x[n-1] + b* x[n-2];
                }
                for(int n = 0 ; n < size; n++){
                    x[n] = y[n];
                }
            }
        }else {
            double sol[10];
            leastsquarepoly(x,sol,size);
            for (int i=0 ; i < size; i++){
                float t = i/1000.0;
                x[i] = -x[i]+sol[0]+(sol[1]*t)+(sol[2]*t*t)+(sol[3]*t*t*t);
            }
        }
    }
}

float getstddev(float* array,int n)
{
    int x;
    float mean=0, Sx2=0, SD;
    for(x = 2; x < n; x++){
        Sx2 += array[x]*array[x];
        mean += array[x];
    }
    mean /= (float)n;

    SD = sqrt(Sx2/n - mean*mean);

    return SD;
}

```

Image.cpp
Main source code for extraction of washboarding profiles

```
float getrms(float* array,int n)
{
    float rms, rms_2 = 0;

    for(int i=0; i< n; i++){
        rms_2 += array[i]*array[i];
    }

    rms = sqrt (rms_2);
    rms /= float (n+1);

    return rms;
}

float getmax(float* array,int n)
{
    float max;

    max = array[0];
    for(int i= 1; i< n; i++){
        if (array[i] > max) max = array[i];
    }

    return max;
}

float getaveamp(float* array,int n)
{
    float max;
    //int i = (20*n)/100;
    int i = 0;
    max = array[i];
    for(i++; i<n/*i< (80*n)/100*/; i++){
        if (array[i] > max) max = array[i];
    }

    return max;
}

float getavemin(float* array,int n)
{
    float min;
    int i = 0;
    //int i = (20*n)/100;
    min = array[i];
    for(i++; i< n/*i< (80*n)/100*/; i++){
        if (array[i] < min) min = array[i];
    }

    return min;
}

float TDrawWindow::getpeaks(float* array,int n, float &amplitude)
{
    const int SAMPLESIZE = 160;

    if (n < SAMPLESIZE) {return -1;} // error n must be bigger than 100

    float* rpart;
    float* ipart;

    try {
        rpart = (float*) new float [512];
        ipart = (float*) new float [512];
    }
    catch (xalloc) { // ENTER THIS BLOCK ONLY IF xalloc IS THROWN.
        // YOU COULD REQUEST OTHER ACTIONS BEFORE TERMINATING
        return 69;
    }

    int l;
    for(l =0; l < n; l++){
        rpart[l] = array[l];
        ipart[l] = 0.0;
    }

    fft(rpart, ipart, SAMPLESIZE, SAMPLESIZE, SAMPLESIZE, 1);

    float max, tmp;
    int where;
    float period;
    max = rpart[2]*ipart[2]; // avoid dc part
    where = 2;
    amplitude = 0;
    for ( int i = 3; i < SAMPLESIZE/2; i++){
        tmp = rpart[i]*rpart[i]; //+ipart[i]*ipart[i];      temp
        if (max < tmp){ // find where max freq is
            max = tmp;
            where = i;
        }
        amplitude += tmp;
    }

    period =((float(where)/float(SAMPLESIZE)) * float(n)); // return freq or peaks
    // amplitude = max;

    if(OptionsBuffer.DoFFTphase == BF_CHECKED){
        TClientDC Specdc(*this);
        int Xo = 10, Yo = 700;
    }
}
```

Image.cpp
Main source code for extraction of washboarding profiles

```

    Specdc.SelectObject(TPen (ClassLib::TColor::Black)); //
    Specdc.MoveTo(Xo ,Yo); // draw spectrum of freq of lines
    for(uint16 x = 0; x < SAMPLESIZE/2; x++){ //
        Specdc.LineTo(Xo + (x*6), Yo - 2* sqrt(rpart[x]*rpart[x] +ipart[x]*ipart[x])); //
    }
    delete[] rpart;
    delete[] ipart;
    return period;
}

void TDrawWindow::InitFFTTreal(uint16 ImageLine, uint16 length)
{
    uint16 inc,inc2;

    for(inc = (uint16) StartSelection.x, inc2=0; inc < StartSelection.x + length; inc++,inc2++){
        FFTreal[inc2] = Image[ 512L*ImageLine + inc];
        FFTimg[inc2] = 0;
    }
}

void TDrawWindow::GiveResults(float* tmpphase, int length, float* speky){

    int l2 = EndSelection.x - StartSelection.x;
    float midfreq =0, normal=0 ;

    float stddev = getstddev(tmpphase,length);
    float rms = getrms(tmpphase,length);
    float max = getaveamp(tmpphase,length) - getavemin(tmpphase,length);
    float amplitude, peaks = getpeaks(tmpphase,length,amplitude);

    float bigamp = FindMaxNum(speky,l2/2);

    for (int i = 0; i< length/2; i++){
        if (speky[i] > (bigamp*(OptionsBuffer.Amp/100.0))){ // at least 85% of max amplitude frequency
            midfreq += float(i) ;
            normal++;
        }
    }
    midfreq /= normal+1;

    float Dlg,Dgb,G,Angle,ImageWidth, Gprime, Lspacing, Tpp, App;

    Dlg = atof(OptionsBuffer.cIw);
    Dgb = atof(OptionsBuffer.cBw);
    G = atof(OptionsBuffer.cBh);
    Angle = atof(OptionsBuffer.aAngle);
    ImageWidth = atof(OptionsBuffer.aImageWidth);
    App = max;
    Lspacing = (ImageWidth)*(1/midfreq)*float(l2)/512.0; // spacing in middle of selection

    if(OptionsBuffer.DoUseDistance == BF_CHECKED){
        Gprime = G*(Dgb/Dlg +1.0); // effective grid spacing
        Angle = (float)atan(double(Gprime/Lspacing));
        Tpp = App * sin(Angle);
    }else if(OptionsBuffer.DoUseAngle == BF_CHECKED){
        Angle = (Angle/180)*M_PI;
        Tpp = App * 27*Lspacing*.2/512;//sin(Angle);
    }

    InfoBuffer.SD =stddev;
    InfoBuffer.RMS=rms;
    InfoBuffer.TN=App;
    InfoBuffer.PEAKS=peaks;
    InfoBuffer.EAngle=Angle*180/M_PI;
    InfoBuffer.ELines=Lspacing;
    InfoBuffer.EDepth=Tpp*1000.0;

    sprintf(Dialog1Buffer.SD,"SD = %.4f",stddev);
    sprintf(Dialog1Buffer.RMS,"RMS = %.4f",rms);
    sprintf(Dialog1Buffer.TN,"App = %.2f pixels",App);
    sprintf(Dialog1Buffer.PEAKS,"Peaks = %.1f",peaks);
    sprintf(Dialog1Buffer.EAngle,"Angle = %.2f degrees",Angle*180/M_PI);
    sprintf(Dialog1Buffer.ELines,"Lspacing = %.1f mm",Lspacing);
    sprintf(Dialog1Buffer.EDepth,"Depth = %4.0f um",Tpp*1000.0);

    if (single == true){
        TTransferDialog* dialog;
        dialog = new TTransferDialog(this, IDD_DIALOG);
        dialog -> Create();
        dialog -> SetCaption(NameOfFile);
    }
}

void TDrawWindow::CheckXLength(void)
{
    if (length < 200){
        MessageBox("Using x = 200", "The x size should be larger",MB_OK);
        length = 200;
        return;
    }
    if(length >= 509){
        length = 512;
    }else{
        while(length%20&&length != 512) length+=1;
    }
}

int TDrawWindow::GetPhaseOfLine(int freq,float* tmpphase)
{

```

Image.cpp
Main source code for extraction of washboarding profiles

```
//float real[512]={0},img[512]={0};
float mag,/* diff ,*/ phase_0 = 0, sign; // phase_0 is a the phase offset

InitFFTreals((uint16) StartSelection.y, (uint16)length); //
fft(FFTreals,FFTimg,length,length,length,1); // initiates the first number in phase
realPhase[0] = calc_phase(FFTreals[freq],FFTimg[freq]); //

tmpphase[0] = realPhase[0];
int index;
int FFTline;

for(index =1,FFTline = StartSelection.y+1; FFTline < EndSelection.y; FFTline ++,index++){
    InitFFTreals((uint16)FFTline, (uint16)length);
    fft(FFTreals,FFTimg,length,length,length,1);
    realPhase[index] = calc_phase(FFTreals[freq],FFTimg[freq]);

        sign = realPhase[index-1] * realPhase [index];
        mag = fabs(realPhase[index-1] + fabs(realPhase [index]));

            if ((sign<0)&&(mag>5)){ // detects jump in phase
                if(realPhase[index-1] <0){ // adjusts accordingly
                    phase_0 -= 2*M_PI;
                }else{
                    phase_0 += 2*M_PI;
                }
            }
        tmpphase[index] = realPhase[index] + phase_0;
    }
return (index);
}

void TDrawWindow::DrawSpectrum(uint16,float *speky)
{
    float ImageXDistance = atof(OptionsBuffer.aImageWidth);
    TClientDC Specdc(*this); // spectrum
    length = EndSelection.x - StartSelection.x; // change to size of rectangle later
    float tmpphase[512]={0};
    for(int i =0 ; i< 512; i++){
        finalPhase[i]=0;
    }
    int MajorFreqMap[512],index;
    // FILE *out;
    // out = fopen("d:/prog/sd.dat","w");

    CheckXLength();
    float max = FindMaxNum(speky,length/2);

    for (int i = 0; i < length/2; i++){
        if (speky[i] > (max*(OptionsBuffer.Amp/100.0))){ // at least 85% of max amplitude frequency
            MajorFreqMap[i] = 1; // offset somwher? dont know wheher!
        }else{
            MajorFreqMap[i] = 0;
        }
    }

    int numoffreq =0 ;

    int countit = 0;
    for(int i =0; i < length/2;i++){
        if (MajorFreqMap[i]) countit++;
    }

    for(int i =0; i < length/2; i++){
        if (MajorFreqMap[i]){

            numoffreq++;
            index = GetPhaseOfLine(i,tmpphase);
            UpdateGauges((numoffreq *100)/ countit);
            for(int i = 0; i < index;i++){
                Specdc.SetPixel(256+int(tmpphase[i]*20),StartSelection.y+i,ClassLib::TColor::LtRed);
                // draw before filter
            }
            filterarray(tmpphase,index);
            for(int i2 =0; i2< 512; i2++){
                finalPhase[i2] += tmpphase[i2];
            }
            for(int i=0; i < index;i++){
                Specdc.SetPixel(256+tmpphase[i]*20,StartSelection.y+i,ClassLib::TColor::Gray); // draw after filter
            }
        }
    }

    UpdateGauges(0);
    FILE* out;
    out = fopen("c:/phase.dat", "at");
    fprintf(out,"START\n");
    for(int i=0; i < index;i++){
        finalPhase[i] /= float(numoffreq);
        Specdc.SetPixel(256.0+finalPhase[i]*20,StartSelection.y+i,ClassLib::TColor::LtYellow); // draw after filter

        fprintf(out,"%f\t%f\n", (i*ImageXDistance)/512.0,finalPhase[i]*CF);
    }
    // GiveResults (finalPhase,index,speky); // does dialog box and stuff within it
    fprintf(out,"END\n");
    fclose(out);
}

//-----
```


Image.cpp
Main source code for extraction of washboarding profiles

```
void TFFTWindow::DrawFFTlines(ClassLib::TRect& Rect)
{
    /* temp open file for save of typical freq resp */

    FILE *out;
    out = fopen("c:/tempfre.dat","at");
    int /*left,right,*/top,bottom;
    top = Rect.Top();
    if (top%2) top--; //set on even boundry
    bottom = Rect.Bottom();
    if (bottom%2) bottom++; // set on even boundry

    if (top >= 20){top -= 20; bottom +=20;} // detect Refresh
    TClientDC Specdc(*this);
    int Xo = 5;
    for (int i=top; (i< bottom) && (i< FFTnumlines); i+=2){
        Specdc.SelectObject(TPen (ClassLib::TColor(i%5))); //
        // draw spectrum of freq of lines

        Specdc.MoveTo(Xo,20 + i); //
        for(uint16 x = 0; x <FFTlength/2; x++){ //
            // cut start
            if (i == top) fprintf(out,"%f\n", (float)FFTlinesArray[top][x]);
            // cut end

            Specdc.LineTo(Xo+ x, 20 - FFTlinesArray[i][x] + i); //
        }
    }
    fclose(out);
}

void TFFTWindow::LoadArray(float array[512][256],long lengthX,int numlines)
{
    for (int i=0; i< numlines; i++){
        for(long x = 0; x < lengthX/2; x++){
            FFTlinesArray[i][x] = array[i][x];
        }
    }
    FFTlength = lengthX;
    FFTnumlines = numlines;
}

void TFFTWindow::Paint(TDC& , bool , ClassLib::TRect& c)
{
    DrawFFTlines(c);
}

TFFTWindow::TFFTWindow(TWindow* parent, const char far* title =0,TWindow* clientWnd =0, bool shrink = false)
    :TFrameWindow(parent, title, clientWnd, shrink),TWindow(parent,title)
{
    // FFTlinesArray = (int**) new int [512][256];
}

TFFTWindow::~TFFTWindow()
{
    // delete[] FFTlinesArray;
}

int TDrawWindow::FindDominantFreq(int *majfreq, float* MeanSpectrum)
{
    length = EndSelection.x - StartSelection.x; // change to size of rectangle later
    float array[512][256];
    int numlines =EndSelection.y - StartSelection.y;

    tmpwin = new TFFTWindow(this,"FFT of lines");

    tmpwin-> Attr.Style = 0;
    tmpwin-> Attr.Style |= WS_POPUP|WS_CLIPSIBLINGS|WS_SYSMENU|WS_CAPTION| WS_BORDER |WS_VISIBLE;
    tmpwin-> Attr.X = 540;
    tmpwin-> Attr.Y = 20;
    tmpwin-> Attr.W = length/2 +20;
    tmpwin-> Attr.H = numlines + 50;

    if(OptionsBuffer.DoFFTlines == BF_CHECKED){ // option
        tmpwin->Create();
    }
    float MeanMajFreqs=0,normal =0 ;
    int i=0/*, Xo = 0*/, offset =0 ;

    if(length >= 509){
        length = 512;
    }else if((length%20) >=10){
        while(length%20) length-=1;
    }else{
        while(length%20&&length != 512) length+=1;
    }
    for(int i = 0; i< 512; i++){
        MeanSpectrum[i] = 0;
    }

    for(uint16 line = (uint16) StartSelection.y; line < EndSelection.y; line ++,normal++,i++){
        InitFFTreal((uint16)line,(uint16)length);
        fft(FFTreal,FFTimg,length,length,length,1);

        for(uint16 x =0; x < length/2;x++){
            Phase[x] = 0;
        }
    }
}
```

Image.cpp
Main source code for extraction of washboarding profiles

```

    for(uint16 x =0; x < length/2;x++){
        Phase[x] = (FFTreals[x]*FFTreals[x]); // + FFTimg[x]*FFTimg[x]);
                if(Phase[x]>=ERZERO)          Phase[x] = sqrt(Phase[x])/(length/4); // sqrt of 0 crashes pc
    }
    SmoothPhase((uint16)length); // got rid off b/c of majfrqarray
    MeanMajFreqs += float(WhereIsMaxNum(Phase, (length-8)/2));
    majfreq[i] = WhereIsMaxNum(Phase, length/2 - 10)/+ 4*/; // get major phase componet for each line

    for(uint16 x =1; x < length/2;x++){
        MeanSpectrum[x] += Phase[x] / float (EndSelection.y - StartSelection.y);
    } // ^because of smooth function
//-----
    if((OptionsBuffer.DoFFTlines == BF_CHECKED)&&!(offset%4)){ // option
        for(uint16 x = 0; x < length/2; x++){ //
            array[i][x] = Phase[x]; //
        }
    }

//-----
    UpdateGauges((i*100)/(EndSelection.y - StartSelection.y));
    }
    UpdateGauges(0);

    if((OptionsBuffer.DoFFTlines == BF_CHECKED)&&!(offset%4)){ // option
        ClassLib:TRect tmp(0,0,512,512); // safe for number greater the length
        tmpwin -> LoadArray(array, length, numlines);
        tmpwin -> DrawFFTlines(tmp);
    }

    return int (MeanMajFreqs/normal)/+ 4*/; // an offset of 4 because of how the
}

float TDrawWindow::FindMaxNum(float *array,int size){
    float max;
    int n;
    for(max= array[1], n=1; n < size; n++){ // fin max # in array.
        if(array[n] > max) max = array[n]; // Used for plotting the array
    }
    return max;
}

float TDrawWindow::FindMinNum(float *array,int size){
    float min;
    int n;
    for(min= array[1], n=1; n < size; n++){
        if(array[n] < min) min = array[n];
    }
    return min;
}

int TDrawWindow::WhereIsMinNum(float *array,int size){
    float min,where;
    int n;
    for(min= array[1], n=1; n < size; n++){
        if(array[n] < min){
            min = array[n];
            where = n;
        }
    }
    return where;
}

int TDrawWindow::WhereIsMaxNum(float *array,int size){
    float max,where;
    int n;
    for(max= array[1], n=1; n < size; n++){
        if(array[n] > max){
            max = array[n];
            where = n;
        }
    }
    return where;
}

void TDrawWindow::InitScreenImage(void)
{
    TPalette MonoPal(MyLogPalette);
    VDC->SelectObject(MonoPal);
    VDC->RealizePalette();
    StartPal = false;
}

void TDrawWindow::DrawBitmap(void)
{
    //VDC = new TClientDC(*this);
    for(long i = 0; i < 512L*512L; i++){
        ScreenImage[i] = char(16.0 + ((Image[i] * 239.0)/255.0)); // to exclude windows colours
    }

    VDC = new TClientDC ((GetApplication()->GetMainWindow()->HWindow);

    TBitmap bitmap(*VDC,512,512);
    bitmap.SetBitmapBits(512L*512L, ScreenImage);

    TDibDC dibdc(bitmap);
    dibdc.SelectObject(bitmap);
    dibdc.SetDIBColorTable(0,255,mono);
}

```

Image.cpp

Main source code for extraction of washboarding profiles

```

dibdc.BitBltToScreen(*VDC,0,0,511,511);
if (HaveSelection){
    VDC->DrawFocusRect (StartSelection.x,StartSelection.y,EndSelection.x,EndSelection.y);
}
}

void TDrawWindow::CmOptions()
{
    TOptionsDialog(this,IDD_OPTIONS).Execute();
}
void TDrawWindow::CmAcquireOptions()
{
    TAcquireOptionsDialog(this,ACQUIREOPTIONS).Execute();
}

void TDrawWindow::CmConvolve()
{
    float matrix[3][3] = {{0,0,1}, {2,3,2}, {1,0,0}};
    float *tmpImage, normal, value=0;

    tmpImage = (float*) new float [512*512];
    for (uint32 i =0 ; i < 512L * 512L; i++){
        tmpImage[i] = Image[i];
        // Image[i] = 0;
    }

    uint32 x,y;
    for(x=normal=0; x<3; x++)
        for(y=0;y<3; y++)
            normal += matrix[x][y];

    for(x=1; x<512-1; x++)
        for(y=1;y<512-1; y++){
            for(int i =-1; i< 2; i++){
                for(int k=-1; k< 2; k++){
                    value += (tmpImage[(x+i) + (y+k)*512] * matrix[1+i][1+k]);
                }
            }
            value /= normal;
            if (value > 254) value = 254;
            Image[x+y*512L] = value;
            value = 0; //was one but dont know why?
        }

    delete[] tmpImage;

    DrawBitmap();
}

void TDrawWindow::SmoothPhase(uint16 length)
{
    float matrix[9] = {1,0,0, 0,0,0,0,0,0};
    float *tmpPhase, normal, value;
    uint16 x;

    for(x=0; x< 3; x++) Phase[x] = 0; // get rid of close dc stuff
    tmpPhase = (float*) new float [length];
    for (uint16 i =0 ; i < length; i++){
        tmpPhase[i] = Phase[i];
    }

    for(x=normal=0; x<9; x++)
        normal += matrix[x];

    for(x=0; x<length-9; x++){
        for(int i =value=0; i < 9; i++){
            value += (tmpPhase[x+i] * matrix[i]);
        }
        value /= normal;
        Phase[x] = value;
    }

    delete[] tmpPhase;
}

void TDrawWindow::SmoothArray(float* array ,uint16 &length)
{
    float matrix[9] = {0,0,1, 2,3,2,1,0,0};
    float *tmpPhase, normal, value;
    tmpPhase = (float*) new float [length];
    for (uint16 i =0 ; i < length; i++){
        tmpPhase[i] = array[i];
        // Image[i] = 0;
    }
    uint16 x;
    for(x=normal=0; x<9; x++)
        normal += matrix[x];

    for(x=0; x<length-9; x++){
        for(int i =value=0; i < 9; i++){
            value += (tmpPhase[x+i] * matrix[i]);
        }
        value /= normal;
        array[x] = value;
    }
    length -= uint16(8);
    delete[] tmpPhase;
}

```

Image.cpp
Main source code for extraction of washboarding profiles

```
float TDrawWindow::calc_phase(float a,float b){
    float ph1,phs1;
    const err1=1e-10;
    if (a!=0){
        if (abs(b/a)<err1) b=0;
        ph1=atan(b/a);
    }phs1=ph1;

    if ((b<0)      &&      (a>0))    phs1=ph1+2*M_PI;
    if ((a<0)      && (b!=0))    phs1=ph1+M_PI;
    if ((a==0)     && (b>0))    phs1=M_PI/2;
    if ((a==0)     && (b<0))    phs1=3*M_PI/2;
    if ((b==0)     && (a<0))    phs1=M_PI;
    if ((b==0)     && (a==0))    phs1=M_PI;
    return (phs1-M_PI);
}

bool
TDrawWindow::CanClose()
{
    return MessageBox("Are you sure?", "Quit",
        MB_YESNO | MB_ICONQUESTION) == IDYES;
}

void
TDrawWindow::EvLButtonDown(uint, ClassLib::TPoint& point)
{
    VDC = new TClientDC(*this);
    ButtonDown = true;

    if(WBcreated == true){
        VDC->DrawFocusRect (StartSelection.x,StartSelection.y,EndSelection.x,EndSelection.y);

        StartSelection = point;
        NewSelection = true;
    }
    if(PRcreated == true){
        if(seta == true){
            VDC -> DrawFocusRect (SetA_SPoint.x,SetA_SPoint.y,SetA_EPoint.x,SetA_EPoint.y);
            SetA_SPoint =SetA_EPoint = oldpointA= point;
        }
        if(setb == true){
            VDC -> DrawFocusRect (SetB_SPoint.x,SetB_SPoint.y,SetB_EPoint.x,SetB_EPoint.y);
            SetB_SPoint =SetB_EPoint = oldpointB= point;
        }
    }
}

void
TDrawWindow::EvLButtonUp(uint, ClassLib::TPoint& point)
{
    if(WBcreated == true){
        EndSelection = point;
        HaveSelection = true;
        Invalidate(false);
    }
    if(PRcreated == true){
        if(seta == true){
            SetA_EPoint = point;
        }
        if(setb == true){
            SetB_EPoint = point;
        }
    }
    ButtonDown = false;
}

void
TDrawWindow::EvMouseMove(uint, ClassLib::TPoint& point)
{
    VDC = new TClientDC(*this);
    if(ButtonDown){
        if(WBcreated == true){

            if(NewSelection){
                oldpoint= StartSelection;
                NewSelection = false;
            }else{
                VDC->DrawFocusRect (StartSelection.x,StartSelection.y,oldpoint.x,oldpoint.y);
                VDC->DrawFocusRect (StartSelection.x,StartSelection.y,point.x,point.y);
                oldpoint = point;
            }
        }
        if(PRcreated == true){
            if(seta == true){

                VDC -> DrawFocusRect (SetA_SPoint.x,SetA_SPoint.y,oldpointA.x,oldpointA.y);
                SetA_EPoint = point;
                VDC -> DrawFocusRect (SetA_SPoint.x,SetA_SPoint.y,SetA_EPoint.x,SetA_EPoint.y);
                oldpointA = SetA_EPoint;
            }
            if(setb == true){
                VDC -> DrawFocusRect (SetB_SPoint.x,SetB_SPoint.y,oldpointB.x,oldpointB.y);
                SetB_EPoint = point;
                VDC -> DrawFocusRect (SetB_SPoint.x,SetB_SPoint.y,SetB_EPoint.x,SetB_EPoint.y);
                oldpointB = SetB_EPoint;
            }
        }
    }
}
```

Image.cpp
Main source code for extraction of washboarding profiles

```
    }  
}  
  
void TDrawWindow::Paint(TDC&, bool , ClassLib::TRect& )  
{  
    DrawBitmap();  
}  
  
int findmode(int *array, int length)  
{  
    int number[512]= {0}, i;  
  
    for(i=0;i< length;i++){  
        number[array[i]] += 1;  
    }  
  
    int max,where;  
    int n;  
    for(max= array[1], n=1; n < length; n++){  
        if(number[n] > max){  
            max = number[n];  
            where = n;  
        }  
    }  
    return where;  
}  
  
void  
TDrawWindow::DoIt(void)  
{  
    int *majfreqarray;  
    float *Spectrum;  
  
    majfreqarray = (int*) new int [512];  
    Spectrum = (float*) new float [512];  
  
    /*int majfreq =*/ FindDominantFreq(majfreqarray,Spectrum);  
  
    uint16 line=0;  
  
    /*int mode =*/ findmode(majfreqarray,EndSelection.y- StartSelection.y);  
  
    delete[] majfreqarray;  
    delete[] Spectrum;  
}  
  
void  
TDrawApp::InitMainWindow()  
{  
    TDrawWindow* DrawWindow = new TDrawWindow;  
    // Construct the decorated frame window  
    TDecoratedFrame* frame = new TDecoratedFrame(0,"Washboard Image Manipulation", DrawWindow, true);  
    SetMainWindow(frame);  
    GetMainWindow()->AssignMenu(COMMANDS);  
    GetMainWindow()->Attr.Style = 0;  
    GetMainWindow()->Attr.X = 0;  
    GetMainWindow()->Attr.Y = 0;  
    GetMainWindow()->Attr.W = 520;  
    GetMainWindow()->Attr.H = 712;  
    GetMainWindow()->Attr.Style |=WS_SYSMENU|WS_CAPTION| WS_BORDER |WS_VISIBLE|WS_MINIMIZEBOX;  
    GetMainWindow()->SetBkgndColor(ClassLib::TColor::LtGray);  
  
    // Construct a status bar  
    TStatusBar* sb = new TStatusBar(frame, TGadget::None);  
    frame->Insert(*sb, TDecoratedFrame::Bottom);  
    frame->SetIcon(this, SMICON);  
    frame->SetIconSm(this, SMICON);  
    // Set the main window and its menu  
}  
  
int  
OwlMain(int /*argc*/, char* /*argv*/ [])  
{  
    return TDrawApp().Run();  
}
```

OpenFiles.cpp
Source code for loading previously captured images

```
#define _USE_OWL_CM_EXIT
#define _OVLVCLPCH
#include <owl/pch.h>
#include "Image67forbuilder.h"

void
TDrawWindow::CmOpenMult()
{
    NewImage = true;
    if (PRcreated==true) KillPR();
    if (WBcreated==true) KillWB();

    single = true;
    TFileOpenDialog::TData ImageFile
        (OFN_FILEMUSTEXIST | OFN_HIDEREADONLY | OFN_PATHMUSTEXIST,
         "Image Files (*.IMG)|*.IMG|", 0, "", "");

    while (TFileOpenDialog(this, ImageFile).Execute() == IDOK) {

        FILE *readimage;
        readimage = fopen(ImageFile.FileName, "rb");
        sprintf(NameOfFile, "%s", ImageFile.FileName);
        fread(Image, 1, 512*512, readimage);
        fclose(readimage);

        DrawBitmap();

        HaveSelection = true;
        NewSelection = false;
        StartSelection.x = atoi(OptionsBuffer.X1);
        StartSelection.y = atoi(OptionsBuffer.Y1);
        EndSelection.x = atoi(OptionsBuffer.X2);
        EndSelection.y = atoi(OptionsBuffer.Y2);

        if (OptionsBuffer.DoImageConvolve == BF_CHECKED) CmConvolve();
        DoIt();
    }
}

void
TDrawWindow::CmAutoCalc()
{
    NewImage = true;
    StartPal = true;
    if (PRcreated==true) KillPR();
    if (WBcreated==true) KillWB();

    single = false;
    char FileNum[5];
    TFileOpenDialog::TData ImageFile
        (OFN_FILEMUSTEXIST | OFN_HIDEREADONLY | OFN_PATHMUSTEXIST,
         "Image Files (*.IMG)|*.IMG|", 0, "", "");

    TOpenSaveDialog::TData data
        (OFN_HIDEREADONLY | OFN_OVERWRITEPROMPT,
         "Data Files (*.WB)|*.WB|", 0, "", "WB");

    if (TFileOpenDialog(this, ImageFile).Execute() == IDOK) {
        if (TFileSaveDialog(this, data).Execute() == IDOK) {
            FILE *out;
            if ((out = fopen(data.FileName, "a+t")) == NULL) return;

            RDialog->Create();

            fprintf(out, "SD um\t");
            fprintf(out, "RMS um\t");
            fprintf(out, "Top Num\t");
            fprintf(out, "# peaks\t");
            fprintf(out, "Degrees\t");
            fprintf(out, "line sep.\t");
            fprintf(out, "Depth calc\n");

            HaveSelection = true;
            NewSelection = false;
            StartSelection.x = atoi(OptionsBuffer.X1);
            StartSelection.y = atoi(OptionsBuffer.Y1);
            EndSelection.x = atoi(OptionsBuffer.X2);
            EndSelection.y = atoi(OptionsBuffer.Y2);

            sprintf(NameOfFile, "%s", ImageFile.FileName);

            int filenamelength, index, goback;
            for (index= 0; NameOfFile[index] != NULL; index++){
                filenamelength = index;
            }

            for (index = filenamelength-5; isdigit(NameOfFile[index]); index--){
                goback = index +1;
            }

            for (index =0; index < 3; index++){
                FileNum[index] = NameOfFile[goback + index]; // ...\.xxx.img|
            }
            FileNum[index+1] = NULL;

            NameOfFile[goback] = NULL;
            char tmpname[100];
            strcpy(tmpname, NameOfFile);

            int FNum = atoi(FileNum);
        }
    }
}
```

OpenFiles.cpp
Source code for loading previously captured images

```
FILE *readimage;
bool stop = false;

for(;stop == false;FNum++){

    sprintf(FileNum, "%d", FNum);
    strcpy(NameOfFile, tmpname);
    strcat(NameOfFile, FileNum);
    strcat(NameOfFile, ".img");

    if ((readimage = fopen(NameOfFile, "rb"))==NULL){
        fclose(out);
        return;
    }

    fread(Image, 1, 512*512, readimage);
    fclose (readimage);
    DrawBitmap();

    if(OptionsBuffer.DoImageConvolve == BF_CHECKED) CmConvolve();
    DoIt();

    char tmp[30];
    sprintf(tmp, "%d \t\t", FNum);
    RDialog->Insert(tmp);

    sprintf(tmp, "%f\t\t", InfoBuffer.SD*Cf*3);
    RDialog->Insert(tmp);
    sprintf(tmp, "%f\r\n", InfoBuffer.RMS*Cf);
    RDialog->Insert(tmp);

    fprintf(out, "%f\t", InfoBuffer.SD*Cf);
    fprintf(out, "%f\t", InfoBuffer.RMS*Cf);
    fprintf(out, "%f\t", InfoBuffer.TN);
    fprintf(out, "%f\t", InfoBuffer.PEAKS);
    fprintf(out, "%f\t", InfoBuffer.EAngle);
    fprintf(out, "%f\t", InfoBuffer.ELines);
    fprintf(out, "%f\n", InfoBuffer.EDepth);
}
}
}

void
TDrawWindow::LoadLogo()
{
    FILE *readimage;
    if((readimage = fopen("amcor.raw", "rb"))!=NULL){
        fread(Image, 1, 512*512, readimage);
        fclose (readimage);
        DrawBitmap();

        HaveSelection = false;
        NewSelection = true;
    }else {
        MessageBox("Error Loading Logo", "File Error", MB_OK);
    }
}

void
TDrawWindow::CmFileOpen()
{
    NewImage = true;
    if(PRcreated==true) KillPR();
    if(WBcreated==true) KillWB();
    SetA_SPoint =SetA_EPoint = oldpointA;
    SetB_SPoint =SetB_EPoint = oldpointB;

    single = true;

    TFileOpenDialog::TData ImageFile
        (OFN_FILEMUSTEXIST | OFN_HIDEREADONLY | OFN_PATHMUSTEXIST,
        "Image Files (*.IMG)|*.IMG|", 0, "", "*");

    if (TFileOpenDialog (this, ImageFile).Execute() == IDOK) {

        FILE *readimage;
        readimage = fopen(ImageFile.FileName, "rb");
        sprintf(NameOfFile, "%s", ImageFile.FileName);
        fread(Image, 1, 512*512, readimage);
        fclose (readimage);
        DrawBitmap();

        HaveSelection = false;
        NewSelection = true;
        StartSelection.x =atoi(OptionsBuffer.X1);
        StartSelection.y =atoi(OptionsBuffer.Y1);
        EndSelection.x =atoi(OptionsBuffer.X2);
        EndSelection.y =atoi(OptionsBuffer.Y2);
    }else {
        MessageBox("Error Loading Image", "File Error", MB_OK);
    }
}
}
```

Source code for Acquiring an image from a CCD camera using the Data Translation PCI card

```

/* source for image acquire by Sven D. Wendler*/
#include <limits.h>
#include <windows.h>
#include "olwintyp.h"
#include "olimgapi.h"
#include "olfgapi.h"

OLT_APISTATUS Status;
int iCount;
OLT_IMGDEVINFO DevInfoList;
char* Alias;
OLT_IMG_DEV_ID DeviceId;
OLT_FG_FRAME_ID FrameId;
OLT_FG_FRAME_INFO FrameInfo;
USHRT NewSource = 0, OldSource;
bool AcqSet;
int VideoSource = 2;

typedef unsigned char uint8;
int SetLevels(long WhiteLevel, long BlackLevel)
{
#ifdef HAVEBOARD
    ULONG old;
    if ((Status = OlFgSetInputControlValue(DeviceId, NewSource, OLC_FG_CTL_WHITE_LEVEL, WhiteLevel, &old)) != OLC_STS_NORMAL) {
        if (DeviceId != NULL) OlImgCloseDevice(DeviceId);
        return 17;
    }

    if ((Status = OlFgSetInputControlValue(DeviceId, NewSource, OLC_FG_CTL_BLACK_LEVEL, BlackLevel, &old)) != OLC_STS_NORMAL) {
        if (DeviceId != NULL) OlImgCloseDevice(DeviceId);
        return 18;
    }
#endif
}

int SetAcqDev(void)
{
#ifdef HAVEBOARD
    NewSource = VideoSource;

    if ((Status = OlImgGetDeviceCount(&iCount)) != OLC_STS_NORMAL) {
        return 1;
    }
    DevInfoList.StructSize = sizeof(OLT_IMGDEVINFO);
    Alias = (char*) &DevInfoList.Alias;

    if ((Status = OlImgGetDeviceInfo(&DevInfoList, sizeof(OLT_IMGDEVINFO))) != OLC_STS_NORMAL) {
        return 2;
    }

    if (!OlImgIsOkay(OlImgOpenDevice(Alias, &DeviceId))) {
        if (DeviceId != NULL) OlImgCloseDevice(DeviceId);
        return 3;
    }

    if ((Status = OlFgSetInputVideoSource(DeviceId, NewSource, &OldSource)) != OLC_STS_NORMAL) {
        if (DeviceId != NULL) OlImgCloseDevice(DeviceId);
        return 4;
    }

    ULONG old;
    if ((Status = OlFgSetInputControlValue(DeviceId, NewSource, OLC_FG_CTL_CSYSNCSHRESH, (ULONG) 125, &old)) != OLC_STS_NORMAL) {
        if (DeviceId != NULL) OlImgCloseDevice(DeviceId);
        return 15;
    }
    ULONG ulSource = MAKELONG(OLC_FG_CSYSNCSHRESH_SPECIFIC_SRC, NewSource);
    if ((Status = OlFgSetInputControlValue(DeviceId, NewSource, OLC_FG_CTL_CSYSNCSHRESH_SOURCE, ulSource, &old)) != OLC_STS_NORMAL) {
        if (DeviceId != NULL) OlImgCloseDevice(DeviceId);
        return 16;
    }

    if ((Status = OlFgAllocateBUILTInFrame(DeviceId, OLC_FG_DEV_MEM_VOLATILE, 1, &FrameId)) != OLC_STS_NORMAL) {
        OlFgDestroyFrame(DeviceId, FrameId);
        if (DeviceId != NULL) OlImgCloseDevice(DeviceId);
        return 5;
    }
    FrameInfo.StructSize = sizeof(OLT_FG_FRAME_INFO);
    if ((Status = OlFgMapFrame(DeviceId, FrameId, &FrameInfo)) != OLC_STS_NORMAL) {
        OlFgDestroyFrame(DeviceId, FrameId);
        if (DeviceId != NULL) OlImgCloseDevice(DeviceId);
        return 6;
    }
    return 0;
#endif
}

void KillAcqDev(void)
{
#ifdef HAVEBOARD
    AcqSet = false;
    OlFgDestroyFrame(DeviceId, FrameId);
    if (DeviceId != NULL) OlImgCloseDevice(DeviceId);
#endif
}

int LoadBuffer(int Width, int Height, uint8* cBuffer)
{

```


Source code for Acquiring an image from a CCD camera using the Data Translation PCI card

```
#ifndef HAVEBOARD
int result;
if (AcqSet== false){
    AcqSet= true;
    result = SetAcqDev();
    if (result != 0) return result;
}

if ((Status = OlFgAcquireFrameToDevice(DeviceId,FrameId/*,&ImageBuffer,768*576*/))!=OLC_STS_NORMAL){

return 7;
}
if ((Status = OlFgReadFrameRect(DeviceId,FrameId,0,0,Width,Height,(uint8*)cBuffer,512L*512L))!=OLC_STS_NORMAL){

    return 8;
}

return 0;
#endif
}
```

Filter.h
Header for filter program dialog box

```
//-----  
#ifndef FilterH  
#define FilterH  
namespace StdCtrls(  
//-----  
#include <Classes.hpp>  
#include <Controls.hpp>  
#include <StdCtrls.hpp>  
#include <Forms.hpp>  
#include <ExtCtrls.hpp>  
#include <ComCtrls.hpp>  
  
//-----  
class TFilterForm : public TForm  
{  
    __published:        // IDE-managed Components  
        TRadioGroup *RadioGroup1;  
        TRadioButton *RadioButton1;  
        TRadioButton *RadioButton2;  
        TRadioButton *RadioButton3;  
        TRadioButton *CustomButterworth;  
        TRadioButton *RadioButton6;  
        TEdit *FreqLowEdit;  
        TEdit *FreqHighEdit;  
        TEdit *RippleEdit;  
        TStaticText *StaticText2;  
        TStaticText *StaticText3;  
        TStaticText *StaticText5;  
        TTrackBar *TrackBar1;  
        TTrackBar *TrackBar2;  
        TButton *Button1;  
        TRadioButton *RadioButton4;  
        TRadioButton *RadioButton5;  
        TRadioButton *RadioButton7;  
        TStaticText *Custom;  
        TStaticText *StaticText1;  
        TStaticText *STFreqCutOff;  
        void __fastcall TrackBar1Change(TObject *Sender);  
        void __fastcall TrackBar2Change(TObject *Sender);  
        void __fastcall RadioButton3Click(TObject *Sender);  
        void __fastcall Button1Click(TObject *Sender);  
        void __fastcall RippleEditEnter(TObject *Sender);  
        void __fastcall RadioButton1Click(TObject *Sender);  
        void __fastcall RadioButton2Click(TObject *Sender);  
        void __fastcall CustomButterworthClick(TObject *Sender);  
        void __fastcall RadioButton6Click(TObject *Sender);  
  
        void __fastcall RadioButton7Click(TObject *Sender);  
        void __fastcall RadioButton4Click(TObject *Sender);  
        void __fastcall RadioButton5Click(TObject *Sender);  
        void __fastcall FreqLowEditEnter(TObject *Sender);  
        void __fastcall FreqHighEditEnter(TObject *Sender);  
private: // User declarations  
public: // User declarations  
    enum FilterTypes { None,LSQ , Normal, PreDefined2 ,LowPass, HighPass, Butt, Cheb} Filter;  
    float FreqLow, FreqHigh, Ripple;  
    __fastcall TFilterForm(TComponent* Owner);  
};  
//-----  
extern PACKAGE TFilterForm *FilterForm;  
//-----  
}  
#endif
```

Filter.cpp
Source code for filter program dialog box

```
//-----  
#include <vcl.h>  
#pragma hdrstop  
  
#include <math.h>  
#include "Filter.h"  
//-----  
#pragma package(smart_init)  
#pragma resource "*.dfm"  
TFilterForm *FilterForm;  
//-----  
__fastcall TFilterForm::TFilterForm(TComponent* Owner)  
: TForm(Owner)  
{  
    FreqLow = TrackBar1->Position;  
    FreqLowEdit->Text = FreqLow;  
    FreqHigh = TrackBar2->Position;  
    FreqHighEdit->Text = FreqHigh;  
    Ripple = 0.1;  
    RippleEdit->Text = Ripple;  
    Filter = Normal;  
    RippleEdit -> Enabled = false;  
    FreqLowEdit -> Enabled = false;  
    TrackBar1 -> Enabled = false;  
    FreqHighEdit -> Enabled = false;  
    TrackBar2 -> Enabled = false;  
    StaticText2 -> Enabled = false; //freqlow  
    StaticText3 -> Enabled = false; //freqhigh  
    StaticText5 -> Enabled = false; //ripple  
    STFreqCutOff -> Enabled = false;  
}  
//-----  
  
void __fastcall TFilterForm::TrackBar1Change(TObject *Sender)  
{  
    FreqLow = TrackBar1->Position;  
    FreqLowEdit->Text = FreqLow;  
    if ((FreqLow >= FreqHigh) && FreqHighEdit -> Enabled == true ){  
        FreqLow = FreqHigh-1;  
        FreqLowEdit->Text = FreqLow;  
        TrackBar1->Position = FreqLow;  
    }  
}  
//-----  
  
void __fastcall TFilterForm::TrackBar2Change(TObject *Sender)  
{  
    FreqHigh = TrackBar2->Position;  
    FreqHighEdit->Text = FreqHigh;  
    if (FreqHigh <= FreqLow){  
        FreqHigh = FreqLow +1;  
        FreqHighEdit->Text = FreqHigh;  
        TrackBar2->Position = FreqHigh;  
    }  
}  
//-----  
  
void __fastcall TFilterForm::RadioButton3Click(TObject *Sender)  
{  
    Filter = LSQ;  
    RippleEdit -> Enabled = false;  
    FreqLowEdit -> Enabled = false;  
    TrackBar1 -> Enabled = false;  
    FreqHighEdit -> Enabled = false;  
    TrackBar2 -> Enabled = false;  
    StaticText2 -> Enabled = false; //freqlow  
    StaticText3 -> Enabled = false; //freqhigh  
    StaticText5 -> Enabled = false; //ripple  
    STFreqCutOff -> Enabled = false;  
}  
//-----  
  
void __fastcall TFilterForm::Button1Click(TObject *Sender)  
{  
    Close();  
}  
//-----  
  
void __fastcall TFilterForm::RippleEditEnter(TObject *Sender)  
{  
    Ripple = atof(RippleEdit->Text.c_str());  
    if(Ripple < 0.0000 || Ripple > 0.9999){  
        Ripple = 0.1;  
        RippleEdit->Text = Ripple;  
    }  
}  
//-----  
  
void __fastcall TFilterForm::RadioButton1Click(TObject *Sender)  
{  
    Filter = Normal;  
    RippleEdit -> Enabled = false;  
    FreqLowEdit -> Enabled = false;  
    TrackBar1 -> Enabled = false;  
    FreqHighEdit -> Enabled = false;  
    TrackBar2 -> Enabled = false;  
    StaticText2 -> Enabled = false; //freqlow  
    StaticText3 -> Enabled = false; //freqhigh
```

Filter.cpp
Source code for filter program dialog box

```
    StaticText5 -> Enabled = false; //ripple
    STFreqCutOff -> Enabled = false;
}
//-----

void __fastcall TFilterForm::RadioButton2Click(TObject *Sender)
{
    Filter = PreDefined2;
    RippleEdit -> Enabled = false;
    FreqLowEdit -> Enabled = false;
    TrackBar1 -> Enabled = false;
    FreqHighEdit -> Enabled = false;
    TrackBar2 -> Enabled = false;
    StaticText2 -> Enabled = false; //freqlow
    StaticText3 -> Enabled = false; //freqhigh
    StaticText5 -> Enabled = false; //ripple
    STFreqCutOff -> Enabled = false;
}
//-----

void __fastcall TFilterForm::CustomButterworthClick(TObject *Sender)
{
    Filter = Butt;
    RippleEdit -> Enabled = false;
    FreqLowEdit -> Enabled = true;
    TrackBar1 -> Enabled = true;
    FreqHighEdit -> Enabled = true;
    TrackBar2 -> Enabled = true;
    StaticText2 -> Enabled = true; //freqlow
    StaticText3 -> Enabled = true; //freqhigh
    StaticText5 -> Enabled = false; //ripple
    STFreqCutOff -> Enabled = true;
}
//-----

void __fastcall TFilterForm::RadioButton6Click(TObject *Sender)
{
    Filter = Cheb;
    RippleEdit -> Enabled = true;
    FreqLowEdit -> Enabled = true;
    TrackBar1 -> Enabled = true;
    FreqHighEdit -> Enabled = true;
    TrackBar2 -> Enabled = true;
    StaticText2 -> Enabled = true; //freqlow
    StaticText3 -> Enabled = true; //freqhigh
    StaticText5 -> Enabled = true; //ripple
    STFreqCutOff -> Enabled = true;
}
//-----

void __fastcall TFilterForm::RadioButton7Click(TObject *Sender)
{
    Filter = None;
    RippleEdit -> Enabled = false;
    FreqLowEdit -> Enabled = false;
    TrackBar1 -> Enabled = false;
    FreqHighEdit -> Enabled = false;
    TrackBar2 -> Enabled = false;
    StaticText2 -> Enabled = false; //freqlow
    StaticText3 -> Enabled = false; //freqhigh
    StaticText5 -> Enabled = false; //ripple
    STFreqCutOff -> Enabled = false;
}
//-----

void __fastcall TFilterForm::RadioButton4Click(TObject *Sender)
{
    Filter = LowPass;
    RippleEdit -> Enabled = false;
    FreqLowEdit -> Enabled = true;
    TrackBar1 -> Enabled = true;
    FreqHighEdit -> Enabled = false;
    TrackBar2 -> Enabled = false;
    StaticText2 -> Enabled = false; //freqlow
    StaticText3 -> Enabled = false; //freqhigh
    StaticText5 -> Enabled = false; //ripple
    STFreqCutOff -> Enabled = true;
}
//-----

void __fastcall TFilterForm::RadioButton5Click(TObject *Sender)
{
    Filter = HighPass;
    RippleEdit -> Enabled = false;
    FreqLowEdit -> Enabled = true;
    TrackBar1 -> Enabled = true;
    FreqHighEdit -> Enabled = false;
    TrackBar2 -> Enabled = false;
    StaticText2 -> Enabled = false; //freqlow
    StaticText3 -> Enabled = false; //freqhigh
    StaticText5 -> Enabled = false; //ripple
    STFreqCutOff -> Enabled = true;
}
//-----

void __fastcall TFilterForm::FreqLowEditEnter(TObject *Sender)
{
    float History = FreqLow;
    FreqLow = atof(FreqLowEdit->Text.c_str());
    if((FreqLow < 0 || FreqLow > 180 || FreqLow >= FreqHigh) && FreqHighEdit->Enabled==true){
```

```
        FreqLow = History;
        FreqLowEdit->Text = FreqLow;
    }else if (FreqLow < 1 || FreqLow > 179){
        FreqLow = History;
        FreqLowEdit->Text = FreqLow;
    }else {
        TrackBar1->Position = FreqLow;
    }
}
//-----

void __fastcall TFilterForm::FreqHighEditEnter(TObject *Sender)
{
    float History = FreqHigh;
    FreqHigh = atof(FreqHighEdit->Text.c_str());
    if(FreqHigh < 0 || FreqHigh > 180 || FreqHigh <= FreqLow){
        FreqHigh = History;
        FreqHighEdit->Text = FreqHigh;
    }else {
        TrackBar2->Position = FreqHigh;
    }
}
//-----
```

FilterPoles.cpp
Source code for filter

```
//Adaption of Filter pole Programm 21 in
//"Introductory Digital Signal Processing with computer applications" by
//Paul A. Lynn and Wolfgang Fuerst
//Translated from Pascal to "c" and then adapted by Sven D. Wendler

#include <math.h>
#include <stdio.h>
void FilterPoles(int FilterFamily,int Type, int Order, float F_Low, float F_High,
float RippleFrac, float Poles[][2], int* ZeroType, float *Gain)
{

float PR[51], PI[51];
int FF,FT,N,K,J,IR,M,M1,N1,N2,ST;
float F1,D,W1,C1,C2,B1,B2,E,X,A,Y,B;
float T,C3,C4,C5,R,I,B3,F2,F3,F4,F5,FR,FI,GR,AR;
float GI,AI,SR,SI,H1,H2,P1R,P2R,P1I,P2I,TH;
float RZ[20][3],CP[20][3], NRZ , NCP;

for(K=1;K <51; PR[K]=0,PI[K]=0,K++);

FF = FilterFamily;

FT = Type;

N = Order;
if(FT == 3) { N/=2; goto _1;}

F1 = F_Low; ST = 2;
if (FT>1) { F1=180-F1;}
goto _2;

_1: F2 = F_Low ; F3 = F_High; ST = 1;
F1 = F3-F2;

_2: if(FF == 2) { D = RippleFrac;}

//***** FIND BUTTERWORTH/CHEBYSHEV PARAMETERS ~*****
IR=N%2;N1=N+IR;N2=((3*N+IR)/2-1);
W1=3.141593*F1/360;C1=tan(W1);B1=2*C1;C2=C1*C1;
B2=0.25*B1*B1; if (FF==1){goto _3;}
E=1/sqrt(1/((1-D)*(1-D))-1);
X=pow(sqrt(E*E+1)+E,1.0/N);
Y=1/X;A=0.5*(X-Y);E=0.5*(X+Y);
//***** FIND REAL AND IMAGINARY PARTS OF LOW-PASS POLES *****
_3: for (K=N1;K<N2+1;K++){
T=3.141593*(2*K+1-IR)/(2*N);if (FF==1){ goto _4;}
C3=A*C1*cos(T);C4=B*C1*sin(T);C5=(1-C3)*(1-C3)+C4*C4;
R=2*(1-C3)/C5-1;I=2*C4/C5; goto _5;
_4: B3=1-B1*cos(T)+C2;R=(1-C2)/B3;I=B1*sin(T)/B3;
_5: M=(N2-K)*2+1;
PR[M+IR]=R;PI[M+IR]=I;PR[M+IR+1]=R;PI[M+IR+1]=-I;
}
if (IR==0) {goto _8;}
if (FF==1) {goto _6;}
R=2/(1+A*C1)-1;goto _7;
_6: R=(1-B2)/(1+B1+B2);
_7: PR[1]=R;PI[1]=0;
_8: if (FT==3) {goto _10;}
//***** PRINT OUT Z-PLANE ZERO LOCATIONS *****
if (FT>1){ goto _9;}

NRZ = 1 ; RZ[1][1] = -1 ; RZ[1][2] = N;
*ZeroType = 1;
goto _13;

_9: for (M=1;M< N+1;M++){
PR[M]=-PR[M];
}
*ZeroType = 2;
NRZ = 1 ; RZ[1][1] = 1 ; RZ[1][2] = N;
goto _13;

_10:
*ZeroType = 3;
NRZ = 2 ; RZ[1][1] = 1 ; RZ[1][2] = N;
RZ[2][1] = -1 ; RZ[2][2] = N;
//***** LOW-PASS TO BANDPASS TRANSFORMATION *****
F4=F2*3.141593/360;F5=F3*3.141593/360;
A=cos(F4+F5)/cos(F5-F4);
for (M1=0; M1 < 24+1;M1++){
M=1+2*M1;
AR=PR[M];AI=PI[M];
if (fabs(AI)<0.0001){ goto _11;}
FR=A*0.5*(1+AR);FI=A*0.5*AI;
GR=FR*FR-FI*FI-AR;GI =2*FR*FI-AI;
SR=sqrt(fabs(GR+sqrt(GR*GR+GI*GI))/2);SI=GI/(2*SR);
P1R=FR+SR;P1I=FI+SI;P2R=FR-SR;P2I=FI-SI;goto _12;
_11: H1=A*(1+AR)/2;H2=H1*H1-AR;
if (H2>0) {
P1R=H1+sqrt(H2);P2R=H1-sqrt(H2);
P1I=0;P2I=0;goto _12;
}
P1R=H1;P2R=H1;P1I=sqrt(fabs(H2));P2I=-P1I;
_12: PR[M]=P1R;PR[M+1]=P2R;PI[M]=P1I;PI[M+1]=P2I;
}

_13:
NCP = Order/2;
int index=1;
for (J=1; J< N+1;J+=ST,index++){
```

```

__14:  M=J; if (IR==0){goto __15;}
        if (J==2) {M=N+1;}
__15:  R=sqrt(PR[M]*PR[M]+PI[M]*PI[M]);
        TH=atan(fabs(PI[M])/fabs(PR[M]))*180/3.141593;
        if (PR[M]<0){TH=180-TH;};

        CP[index][1]= Poles[index-1][0] = R ; CP[index][2] = TH;
        Poles[index-1][1]= TH*M_PI/180;
__16:}
// ***** Magnitude of gain
int i;
float BASE, EX, W,S1,S2,H[3200];
__4:for (i=1; i<3200;i++)
{
    W=3.141593*(i-1)/3200;
    C1=cos(W);C2=cos(2*W);S1=sin(W);S2=sin(2*W);H[i]=1;

__5:  if (NRZ==0){goto __6;}
        for (K=1; K< NRZ+1;K++){
            A=RZ[K][1];B=RZ[K][2];BASE=1-(2*A*C1)+(A*A);EX=B/2;
            H[i]=H[i]*pow(BASE,EX);
        }
__6:  if (NCP==0){goto __7;}
        for (K=1; K< NCP+1;K++){
            R=CP[K][1];T=CP[K][2]*3.141593/180;
            D=C2-2*R*cos(T)*C1+R*R;E=S2-2*R*cos(T) *S1;
            H[i]=H[i]/pow(D*D+E*E,0.5);
        }
__7:  /* if (NCZ=0) {goto __8;}
        for (K=1; K< NCZ+1;K++){
            R=CZ[K,1];T=CZ[K,2]*3.141593/180;
            D=C2-2*R*cos(T)*C1+R*R;E=S2-2*R*cos(T)*S1;
            H[i]=H[i]/(exp(0.5*log(D*D+E*E)));
        } */
__8:  }

    float max = 0;
    // FILE *out;
    // out = fopen("c:/tmp.dat","wt");
    for (int i=2; i< 3195;i++){
        if (fabs(H[i]) > max) max = fabs(H[i]);
    //    fprintf(out,"%f\n",H[i]);
    }
    /*    printf("Gain = %f, %f",max, 20*log(max)*0.4343);

        fclose(out);*/
    *Gain = max;
}

```

Source code for calculating the a polynomial least squares curve fit of a washboarding profile

```

// This source code for calculating a Least squares curve fit of a washboarding profile maybe used when a filter is not desired.
// A washboarding profile is changed by subtractiong the polynomial fit.
#include <stdio.h>
#include <conio.h>
#include <math.h>
#include <stdlib.h>
#define SIZE 512
double mypow(double a, double b)
{
    if (b< .000000000000000001) return (1.000000000000000);
    return pow(a,b);
}

double det2(double n2[10][10])
{
    return ((n2[0][0] * n2[1][1]) - (n2[0][1] * n2[1][0]));
}

double detn(double det[10][10],int n)
{
    int x, i,j,cnt;
    double tmp[10][10], value = 0;

    for(x=0;x<n;x++){
        for(cnt=0, i=0; i<n-1; cnt++,i++){
            if(cnt==x) cnt++;
            for(j=0;j<n-1;j++){
                tmp[j][i] = det[j+1][cnt];
                // printf("i=%d,j=%d,x=%d,cnt =%d,tmp[j][i]=%f\n",i,j,x,cnt,tmp[j][i]);
            }
            if (n==3) value += mypow(-1,x+1) * det[0][x] * det2(tmp);
            else value += mypow(-1,x+1) * det[0][x] * detn(tmp,n-1);
        }
        // printf("value = %f\n",value);
        return value;
    }
}

void polyn(double *solution, int degree, int nopnts, double *xn, double *yn)
{
    double det[10][10] , sol[10][10];
    int i,j,n,y,x;

    for(x = 0; x< degree; x++){
        for(i = 0; i< degree; i++){
            for(j = 0; j< degree; j++){
                for(n=det[j][i]=sol[j][i] = 0; n< nopnts;n++){
                    det[j][i] = sol[j][i] += mypow(xn[n],j+1);
                }
            }
            for(y = 0; y< degree; y++){
                for(n= sol[y][x]=0; n< nopnts; n++){
                    sol[y][x] += (mypow(xn[n],y)* yn[n]) ;
                }
            }
            solution[x] = detn(sol,degree) / detn(det,degree);
        }
    }
}

void leastsquarepoly(float* array,double* sol,int n)
{
    double chi, xn[SIZE],yn[SIZE], cnt, delta2 = 0, amp;
    int i, sigmahere, inc, inc2,degree = 4;

    for (inc = 0; inc < n; inc++){
        xn[inc] = inc/1000.0;
        yn[inc] = array[inc];
    }

    polyn(sol,degree, n-1, xn,yn);
}

```


SingFFT32.c
Source code for fast Fourier transformation

```

/*****

Javier Soley, Ph. D,   FJSOLEY @UCRVM2.BITNET
Escuela de Física y Centro de Investigaciones Geofísicas
Universidad de Costa Rica

*****/

/*      Computes the DISCRETE FOURIER TRANSFORM of very long data series.
 *      Restriction: the data has to fit in conventional memory.
 *
 *      Compile in compact or large models ==> Pointers must be FAR
 *      Two huge pointers are used to access the real and imaginary
 *      parts of the transform without 64k wrap around.
 *
 *      This functions are translations from the fortran program in
 *
 *      R. C. Singleton, An algorithm for computing the mixed radix fast
 *      Fourier transform
 *
 *      IEEE Trans. Audio Electroacoust., vol. AU-17, pp. 93-10, June 1969.
 *      Some features are:
 *
 *      1-) Accepts an order of transform that can be factored not only
 *           in prime factors such 2 and 4, but also including odd factors
 *           as 3, 5, 7, 11, etc.
 *      2-) Generates sines and cosines recursively and includes
 *           corrections for truncation errors.
 *      3-) The original subroutine accepts multivariate data. This
 *           translation does not implement that option (because I
 *           do not needed right now).
 *
 *      Singleton wrote his subroutine in Fortran and in such a way that it
 *      could be ported almost directly to assembly language. I transcribed
 *      it to C with little effort to make it structured. So I apologize to
 *      all those C purists out there!!!!!!!
 *
 */

/* Version 2.0 March/30/92 */

/* Includes */

#include <stdlib.h>
#include <stdio.h>
#include <math.h>

/* Defines */

#define TWO_PI ((double)2.0 * M_PI)
#define MAXF 23
#define MAXP 209

#define MY_SIZE float /* or float or long double */

/* Globals */

long nn, m, flag, jf, jc, kspan, ks, kt, nt, kk, i;
MY_SIZE c72, s72, s120, cd, sd, rad, radf, at[23], bt[23];
long nfac[23];
int inc;
long np[MAXP];

/* The functions */

void radix_2(MY_SIZE a[1024],MY_SIZE b[1024])
{ long k1, k2;
  MY_SIZE ak, bk, c1, s1;
  kspan >= 1;
  k1 = kspan +2;

  do {
    do {
      k2 = kk + kspan;
      ak = a[k2-1];
      bk = b[k2-1];
      a[k2-1] = a[kk-1] -ak;
      b[k2-1] = b[kk-1] -bk;
      a[kk-1] += ak;
      b[kk-1] += bk;
      kk = k2 + kspan;
    } while ( kk <= nn);
    kk = kk - nn;
  } while ( kk <= jc);

  if ( kk > kspan) flag = 1;
  else
  {
    do {
      c1 = 1.0 - cd;
      s1 = sd;
      do {
        do {
          k2 = kk + kspan;
          ak = a[kk-1]- a[k2-1];
          bk = b[kk-1]- b[k2-1];
          a[kk-1] += a[k2-1];
          b[kk-1] += b[k2-1];
          a[k2-1] = c1*ak - s1*bk;

```

SingFFT32.c
Source code for fast Fourier transformation

```

        b[k2-1] = s1*ak + c1*bk;
        kk = k2 + kspan;
    } while ( kk < nt );

    k2 = kk - nt;
    c1 = -c1;
    kk = k1 - k2;
    } while ( kk > k2 );
    ak = c1 - (cd*c1+sd*s1);
    s1 = (sd*c1-cd*s1) +s1;

/***** Compensate for truncation errors *****/

    c1 = 0.5/(ak*ak+s1*s1)+0.5;
    s1 *= c1;
    c1 *= ak;
    kk += jc;
    } while ( kk < k2);
    k1 = k1 + inc + inc;
    kk = (k1- kspan) /2 + jc;
    } while ( kk <= jc + jc );
}

void radix_4(int isn, MY_SIZE a[1024],MY_SIZE b[1024])
{
    long k1, k2, k3;
    MY_SIZE akp, akm, ajm, ajp, bkm, bkp, bjm, bjp;
    MY_SIZE c1, s1, c2, s2, c3, s3;
    kspan /= 4;

cuatro_1:
    c1 = 1.0;
    s1 = 0;
    do {
        do {
            do {
                k1 = kk + kspan;
                k2 = k1 + kspan;
                k3 = k2 + kspan;
                akp = a[kk-1] + a[k2-1];
                akm = a[kk-1] - a[k2-1];
                ajp = a[ k1-1] + a[k3-1];
                ajm = a[ k1-1] - a[k3-1];
                a[kk-1] = akp + ajp;
                ajp = akp - ajp;
                bkp = b[kk-1] + b[k2-1];
                bkm = b[kk-1] - b[k2-1];
                bjp = b[k1-1] + b[k3-1];
                bjm = b[k1-1] - b[k3-1];
                b[kk-1] = bkp + bjp;
                bjp = bkp - bjp;
                if ( isn < 0) goto cuatro_5;
                akp = akm - bjm;
                akm = akm + bjm;
                bkp = bkm + ajm;
                bkm = bkm - ajm;
                if (s1 == 0.0) goto cuatro_6;
cuatro_3:
                a[ k1-1] = akp*c1 - bkp*s1;
                b[ k1-1] = akp*s1 + bkp*c1;
                a[ k2-1] = ajp*c2 - bjp*s2;
                b[ k2-1] = ajp*s2 + bjp*c2;
                a[ k3-1] = akm*c3 - bkm*s3;
                b[ k3-1] = akm*s3 + bkm*c3;
                kk = k3 + kspan;
            } while ( kk <= nt);

cuatro_4:
            c2 = c1 - (cd*c1 + sd*s1);
            s1 = (sd*c1 - cd*s1) + s1;

/***** Compensate for truncation errors *****/

            c1 = 0.5 / (c2*c2 + s1*s1) +0.5;
            s1 = c1 * s1;
            c1 = c1 * c2;
            c2 = c1*c1 - s1*s1;
            s2 = 2.0 * c1 *s1;
            c3 = c2*c1 - s2*s1;
            s3 = c2*s1 + s2*c1;
            kk = kk -nt + jc;
        } while ( kk <= kspan);

        kk = kk - kspan + inc;
        if ( kk <= jc) goto cuatro_1;
        if ( kspan == jc) flag =1;
        goto out;
cuatro_5:
        akp = akm + bjm;
        akm = akm - bjm;
        bkp = bkm - ajm;
        bkm = bkm + ajm;
        if (s1 != 0.0) goto cuatro_3;
cuatro_6:
        a[k1-1] = akp;
        b[k1-1] = bkp;
        b[k2-1] = bjp;
        a[k2-1] = ajp;
        a[k3-1] = akm;
        b[k3-1] = bkm;
        kk = k3 + kspan;
    } while ( kk <= nt);
    goto cuatro_4;
out:

```

```

    s1 = s1 + 0.0;
}

/* Find prime factors of n */
void fac_des( long n)
{
    long k, j, jj, l;
    k = n;
    m = 0;
    while ( k-(k / 16)*16 == 0 ) {
        m++;
        nfac[m-1] = 4;
        k /= 16;
    }
    j = 3;
    jj = 9;
    do {
        while (k % jj == 0) {
            m++;
            nfac[m-1] = j;
            k /= jj;
        }
        j += 2;
        jj = j * j;
    } while ( jj <= k);
    if (k <= 4) {
        kt = m;
        nfac[m] = k;
        if (k != 1) m++;
    }
    else {
        if (k-(k / 4)*4 == 0) {
            m++;
            nfac[m-1] = 2;
            k /= 4;
        }
        kt = m;
        j = 2;
        do {
            if (k % j == 0) {
                m++;
                nfac[m-1] = j;
                k /= j;
            }
            j = ((j+1) / 2)*2 + 1;
        } while ( j <= k);
    }
    if (kt != 0) {
        j = kt;
        do {
            m++;
            nfac[m-1] = nfac[j-1];
            j--;
        } while ( j != 0);
    }
}

/* Permute the results to normal order */

void permute(long ntot,long n, MY_SIZE a[1024], MY_SIZE b[1024])
{
    long k, j, k1, k2, k3, kspnn, maxf;

    MY_SIZE ak, bk;
    long ii, jj;
    maxf = MAXF;
    np[0] = ks;
    if (kt != 0) {
        k = kt + kt + 1;
        if (m < k) k--;
        j = 1;
        np[k] = jc;
        do {
            np[j] = np[j-1] / nfac[j-1];
            np[k-1] = np[k] * nfac[j-1];
            j++;
            k--;
        } while (j < k);
        k3 = np[k];
        kspan = np[1];
        kk = jc+1;
        k2 = kspan + 1;
        j = 1;
    }

    /***** Permutation of one dimensional transform *****/

    if (n == ntot) {
        do {
            do {
                ak = a[kk-1];
                a[kk-1] = a[k2-1];
                a[k2-1] = ak;
                bk = b[kk-1];
                b[kk-1] = b[k2-1];
                b[k2-1] = bk;
                kk += inc;
                k2 += kspan;
            }
        }
    }
}

```

SingFFT32.c
Source code for fast Fourier transformation

```

ocho_30:                } while ( k2 < ks);
                        do {
                                k2 -= np[j-1];
                                j++;
                                k2 += np[j];
                        } while (k2 > np[j-1]);
                        j = 1;
ocho_40:                j = j + 0;
                        } while (kk < k2);
                        kk += inc;
                        k2 += kspan;
                        if (k2 < ks) goto ocho_40;
                        if (kk < ks) goto ocho_30;
                        jc = k3;
}
ocho_50:                else { /* Permutation for multiple transform */
                        do {
                                do {
                                        k = kk + jc;
                                        do {
                                                ak = a[kk-1];
                                                a[kk-1] = a[k2-1];
                                                a[k2-1] = ak;
                                                bk = b[kk-1];
                                                b[kk-1] = b[k2-1];
                                                b[k2-1] = bk;
                                                kk += inc;
                                                k2 += inc;
                                        } while ( kk < k);
                                        kk = kk + ks - jc;
                                        k2 = k2 + ks - jc;
                                } while (kk < nt);
                                k2 = k2 - nt + kspan;
                                kk = kk - nt + jc;
                        } while (k2 < ks);
                        do {
                                do {
                                        k2 -= np[j-1];
                                        j++;
                                        k2 += np[j];
                                } while (k2 > np[j-1]);
                                j = 1;
                                do {
                                        if ( kk < k2 ) goto ocho_50;
                                        kk +=jc;
                                        k2 += kspan;
                                } while (k2 < ks);
                        } while (kk < ks);
                        jc = k3;
                }
}

if ( (2*kt +1) < m) {
    kspnn = np[kt]; /* Permutation of square-free factors of n */
    j = m - kt;
    nfac[j] = 1;
    do {
            nfac[j-1] *= nfac[j];
            j--;
    } while (j != kt);
    kt++;
    nn = nfac[kt-1] -1;
    if (nn > MAXP) {
            printf("product of square free factors exceeds allowed limit\n");
            exit(2);
    }
    jj =0;
    j=0;
    goto nueve_06;
nueve_02:                jj -= k2;
                        k2 = kk;
                        k++;
                        kk = nfac[k-1];
                        do {
                                jj += kk;
                                if ( jj >= k2 ) goto nueve_02;
                                np[j-1] = jj;
nueve_06:                k2 = nfac[kt-1];
                        k = kt+1;
                        kk = nfac[k-1];
                        j++;
                } while (j <= nn);
                /* determine the permutation cycles of length greater then 1 */
                j =0;
                goto nueve_14;
                do {
                        do {
                                k = kk;
                                kk = np[k-1];
                                np[k-1] = -kk;
                        } while ( kk != j);
                        k3 = kk;
nueve_14:                do {
                                j++;
                                kk = np[j-1];
                        } while (kk <0);
                } while ( kk != j);

```

SingFFT32.c
Source code for fast Fourier transformation

```

np[j-1] = -j;
if (j != nn) goto nueve_14;
maxf *= inc;
/* Reorder a and b following the permutation cycles */
goto nueve_50;
do {
    do {
        do { j--;} while (np[j-1] <0);
        jj = jc;
        do {
            kspan = jj;
            if ( jj > maxf) kspan = maxf;
            jj -= kspan;
            k = np[j-1];
            kk = jc*k + ii + jj;
            k1 = kk + kspan;
            k2 = 0;
            do {
                k2++;
                at[k2-1] = a[k1-1];
                bt[k2-1] = b[k1-1];
                k1 -= inc;
            } while (k1 != kk);
            do {
                k1 = kk + kspan;
                k2 = k1 - jc*(k + np[k-1]);
                k = -np[k-1];
                do {
                    a[k1-1] = a[k2-1];
                    b[k1-1] = b[k2-1];
                    k1 -= inc;
                    k2 -= inc;
                } while (k1 != kk);
                kk = k2;
            } while ( k != j);
            k1 = kk + kspan;
            k2 = 0;
            do {
                k2++;
                a[k1-1] = at[k2-1];
                b[k1-1] = bt[k2-1];
                k1 -= inc;
            } while ( k1 != kk);
        } while ( jj != 0 );
    } while (j !=1);
nueve_50:
    j = k3+1;
    nt -= kspnn;
    ii = nt - inc +1;
    } while ( nt >= 0);
    k = k + 0;
}

/*****
Functions for prime factor radix
*****/
void radix_3(MY_SIZE a[1024],MY_SIZE b[1024])
{
    long k1, k2;
    MY_SIZE ak, bk, aj, bj;

    do {
        do {
            k1 = kk + kspan;
            k2 = k1+ kspan;
            ak = a[kk-1];
            bk = b[kk-1];
            aj = a[k1-1] + a[k2-1];
            bj = b[k1-1] + b[k2-1];
            a[kk-1] = ak + aj;
            b[kk-1] = bk + bj;
            ak = -0.5*aj + ak;
            bk = -0.5*bj + bk;
            aj = (a[k1-1]-a[k2-1])*s120;
            bj = (b[k1-1]-b[k2-1])*s120;
            a[k1-1] = ak - bj;
            b[k1-1] = bk + aj;
            a[k2-1] = ak + bj;
            b[k2-1] = bk - aj;
            kk = k2 + kspan;
        } while ( kk < nn);
        kk = kk - nn;
    } while ( kk <= kspan);
}

void radix_5(MY_SIZE a[1024],MY_SIZE b[1024])
{
    long k1, k2, k3, k4;
    MY_SIZE ak, aj, bk, bj, akp, akm, ajm, ajp, aa, bkp, bkm, bjm, bjp, bb;
    MY_SIZE c2, s2;
    c2 = c72*c72 - s72*s72;
    s2 = 2 * c72 * s72;
    do {
        do {
            k1 = kk + kspan;
            k2 = k1 + kspan;
            k3 = k2 + kspan;

```

```

        k4 = k3 + kspan;
        akp = a[k1-1] + a[k4-1];
        akm = a[k1-1] - a[k4-1];
        bkp = b[k1-1] + b[k4-1];
        bkm = b[k1-1] - b[k4-1];
        ajp = a[k2-1] + a[k3-1];
        ajm = a[k2-1] - a[k3-1];
        bjp = b[k2-1] + b[k3-1];
        bjm = b[k2-1] - b[k3-1];
        aa = a[kk-1];
        bb = b[kk-1];
        a[kk-1] = aa + akp + ajp;
        b[kk-1] = bb + bkp + bjp;
        ak = akp*c72 + ajp*c2 + aa;
        bk = bkp*c72 + bjp*c2 + bb;
        aj = akm*s72 + ajm*s2;
        bj = bkm*s72 + bjm*s2;
        a[k1-1] = ak - bj;
        a[k4-1] = ak + bj;
        b[k1-1] = bk + aj;
        b[k4-1] = bk - aj;
        ak = akp*c2 + ajp*c72 + aa;
        bk = bkp*c2 + bjp*c72 + bb;
        aj = akm*s2 - ajm*s72;
        bj = bkm*s2 - bjm*s72;
        a[k2-1] = ak - bj;
        a[k3-1] = ak + bj;
        b[k2-1] = bk + aj;
        b[k3-1] = bk - aj;
        kk = k4 + kspan;
    } while ( kk < nn);
    kk -= nn;
} while ( kk <= kspan);
}

void fac_imp(MY_SIZE a[1024],MY_SIZE b[1024])
{
    long k, kspnn, j, k1, k2, jj;
    MY_SIZE ak, bk, aa, bb, aj, bj;
    MY_SIZE c1, s1, c2, s2;
    MY_SIZE ck[23], sk[23];

    k = nfac[i-1];
    kspnn = kspan;
    kspan /= k;
    if (k==3) radix_3(a, b);
    if (k==5) radix_5(a, b);
    if ((k==3) || (k==5)) goto twi;
    if (k!=jf) {
        jf = k;
        s1 = rad/k;
        c1 = cos(s1);
        s1 = sin(s1);
        ck[jf-1] = 1.0;
        sk[jf-1] = 0.0;
        j = 1;
        do {
            ck[j-1] = ck[k-1]*c1 + sk[k-1]*s1;
            sk[j-1] = ck[k-1]*s1 - sk[k-1]*c1;
            k--;
            ck[k-1] = ck[j-1];
            sk[k-1] = -sk[j-1];
            j++;
        } while ( j<k);
    }
    do {
        do {
            k1 = kk;
            k2 = kk + kspnn;
            aa = a[kk-1];
            bb = b[kk-1];
            ak = aa;
            bk = bb;
            j = 1;
            k1 = k1 + kspan;
            do {
                k2 -= kspan;
                j++;
                at[j-1] = a[k1-1] + a[k2-1];
                ak += at[j-1];
                bt[j-1] = b[k1-1] + b[k2-1];
                bk += bt[j-1];
                j++;
                at[j-1] = a[k1-1] - a[k2-1];
                bt[j-1] = b[k1-1] - b[k2-1];
                k1 += kspan;
            } while ( k1 < k2);
            a[kk-1] = ak;
            b[kk-1] = bk;
            k1 = kk;
            k2 = kk + kspnn;
            j = 1;
            do {
                k1 += kspan;
                k2 -= kspan;
                jj = j;
                ak = aa;
                bk = bb;
                aj = 0.0;
                bj = 0.0;
                k = 1;

```

SingFFT32.c
Source code for fast Fourier transformation

```

do {
    k++;
    ak = at[k-1]*ck[jj-1] + ak;
    bk = bt[k-1]*ck[jj-1] + bk;
    k++;
    aj = at[k-1]*sk[jj-1] + aj;
    bj = bt[k-1]*sk[jj-1] + bj;
    jj += j;
    if (jj>jf) jj-=jf;
} while ( k<jf);
k = jf - j;
a[k1-1] = ak - bj;
b[k1-1] = bk + aj;
a[k2-1] = ak + bj;
b[k2-1] = bk - aj;
j++;
} while (j<k);
kk += kspnn;
} while (kk <= nn);
kk -= nn;
} while ( kk <= kspan);

/***** Multiply by twiddle factors *****/

twi:
    if (i==m) flag = 1;
    else {
        kk = jc + 1;
        do {
            c2 = 1.0 - cd;
            s1 = sd;
            do {
                c1 = c2;
                s2 = s1;
                kk += kspan;
                do {
                    do {
                        ak = a[kk-1];
                        a[kk-1] = c2*ak - s2*b[kk-1];
                        b[kk-1] = s2*ak + c2*b[kk-1];
                        kk += kspnn;
                    } while ( kk <= nt);
                    ak = s1 * s2;
                    s2 = s1*c2 + c1*s2;
                    c2 = c1*c2 - ak;
                    kk = kk - nt + kspan;
                } while ( kk <= kspnn);
                c2 = c1 - (cd*c1 + sd*s1);
                s1 = s1 + (sd*c1 - cd*s1);

/***** Compensate for truncation errors *****/

                c1 = 0.5/(c2*c2 + s1*s1) + 0.5;
                s1 *= c1;
                c2 *= c1;
                kk = kk - kspnn + jc;
            } while ( kk <= kspan);
            kk = kk - kspan + jc + inc;
        } while ( kk <= (jc+jc));
    }
}

void fft(MY_SIZE a[1024],MY_SIZE b[1024], long ntot,long n,long nspan,int isn)
{
    if ( n < 2 ) exit(1);
    inc = isn;
    rad = TWO_PI;
    s72 = rad / 5.0;
    c72 = cos(s72);
    s72 = sin(s72);
    s120 = sqrt(0.75);
    if (isn < 0) {
        s72 = -s72;
        s120 = -s120;
        rad = -rad;
        inc = -inc;
    }
    nt = inc * ntot;
    ks = inc * nspan;
    kspan = ks;
    nn = nt - inc;
    jc = ks / n;
    radf = rad * jc * 0.5;
    i = 0;
    jf = 0;
    flag = 0;
    fac_des (n );
    do {
        sd = radf / kspan;
        cd = 2.0 * sin(sd) * sin(sd);
        sd = sin(sd+sd);
        kk = 1;
        i = i + 1;
        if (nfac[i-1]==2)
            radix_2( a, b);
        if (nfac[i-1]==4)
            radix_4(isn, a, b);
        if ( ( nfac[i-1]!=2) && (nfac[i-1]!=4))
            fac_imp(a, b);
    } while ( flag != 1);
}

```

SingFFT32.c
Source code for fast Fourier transformation

```

    permute( ntot, n, a, b);
}

/* Calculates the the Fourier transform of 2*half_length real values.
Original data values are stored alternately in arrays a and b.
The cosine coefficients are in a[0], a[1] .....a[half_length] and
the sine coefficients are in b[0], b[1] .....b[half_length].
The coefficients must be scaled by 1/(4*half_length) in the calling
procedure. */

/* April/1/92 Tried Singleton's subroutine and it does not seem to work.
I am modifying it and folowing the procedure of Cooley, Lewis and Welch
J. Sound Vib., vol. 12, pp 315-337, July 1970. Will extend the
procedure so half_length can be odd also. */

/* Assume we have the transform A(n) of x(even) + i x(odd)
1-) A1(n) = (1/2) [ Ac(-n) + A(n)] =(1/2) [Ac(N-n) + A(n)]
(Ac = complex of A) (for N even or odd)
2-) A2(n) = (i/2)[Ac(-n)-A(n)] =(i/2) [Ac(N-n)-A(n)]
(for N even or odd)
3-) C(n) = (1/2)[A1(n) + A2(n)*W2N**(-n)] (1)
0,1,2,3..... N N even
0,1,2,3..... N-1 N odd
Use the simmetry of the A1 and A2 sequences
C(N-n) = (1/2) [A1(N-n) + A2(N-n)*W2N**(-N+n)]
= (1/2) [A1c(n) - A2c(n)*W2N**(n)] (2)
Evaluate (1) and (2) for n=0,1,2,...N/2 -1 and (1) for N/2
if N is even (2) is also good
Evaluate (1) for n =0 and
(1) and (2) for n=1,2,...(N-1)/2
if N is odd

Let the factors of two be taken care in the normalization
outside this procedure, ie, the coefficients will be
four times larger. */

/* 4/april/1992 Everything is working fine. Singleton's Realtr
works ok. */

void realtr(MY_SIZE a[1024],MY_SIZE b[1024],long half_length,int isn)
{
    long nh, j, k;
    MY_SIZE sd, cd, sn, cn, alr, ali, a2r, a2i, re, im;
    nh = half_length >> 1; /* Should work for even and odd */
    sd = M_PI_2 /half_length;
    cd = 2.0 * sin(sd) * sin(sd);
    sd = sin(sd+sd);
    sn = 0;
    if ( isn <0) {
        cn = 1 ;
        a[half_length] = a[0];
        b[half_length] = b[0];
    }
    else {
        cn = -1;
        sd = -sd;
    }
}

/* For nh odd the j = nh value is meaningless (and harmless to calculate).
Also, the value nh /2 might be calculated twice. */

for (j=0; j <= nh; j++) {

    k = half_length-j;
    alr = a[j] + a[k];
    a2r = b[j] + b[k];
    ali = b[j] - b[k];
    a2i = -a[j] + a[k];
    re = cn*a2r + sn*a2i;
    im = -sn*a2r + cn*a2i;
    b[k] = +im - ali;
    b[j] = im + ali;
    a[k] = alr - re;
    a[j] = alr + re;
    alr = cn - (cd*cn + sd*sn);
    sn = (sd*cn - cd*sn) + sn;
    /* compensate for truncation error */
    cn = 0.5/(alr*alr + sn*sn) + 0.5;
    sn *= cn;
    cn *= alr;
}
}

```


Appendix - B

Phase and strain program

This appendix contains the main source code for the program that calculates the strain of a washboarding profile and the shape of washboarding. Figure [B.1] is a screen capture of the program showing the result of calculating the strain from the illustrated profile (see theory section []). Figure [B.2] is a screen capture of the program showing the profile, a selected range (red and blue lines) for Fourier transformation and the result of the transformation of the selected range. The program allows for multiple ranges to be selected, Fourier transformed and then averaged. The full fourier transformation can be saved by pressing the button 'SaveFFT' shown in figure [B.2]. The source code used for Fourier transformation was 'singFFT.c' in appendix A, which was written by R. C. Singleton and translated by Javier Soley.

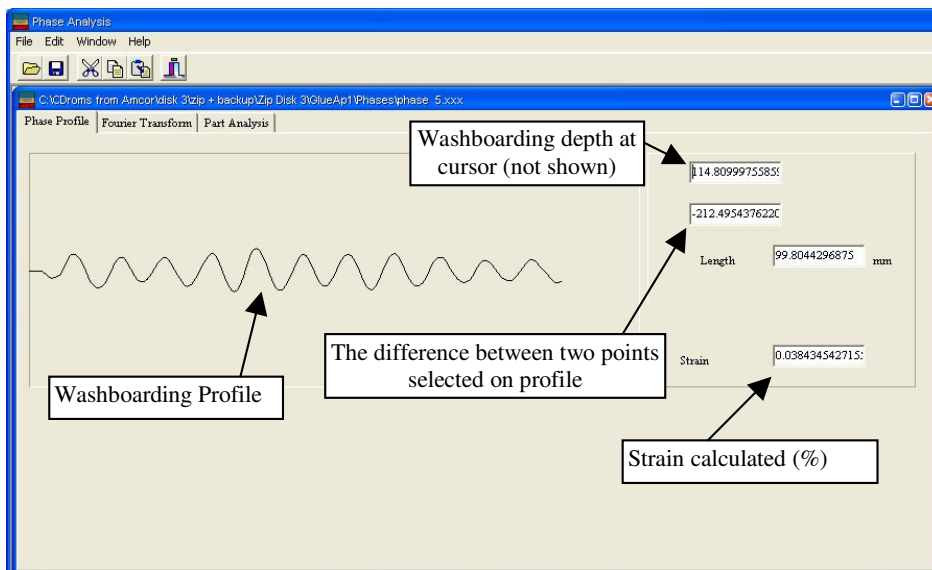


Figure [B.1] – Screen capture of shape and strain program showing strain calculated

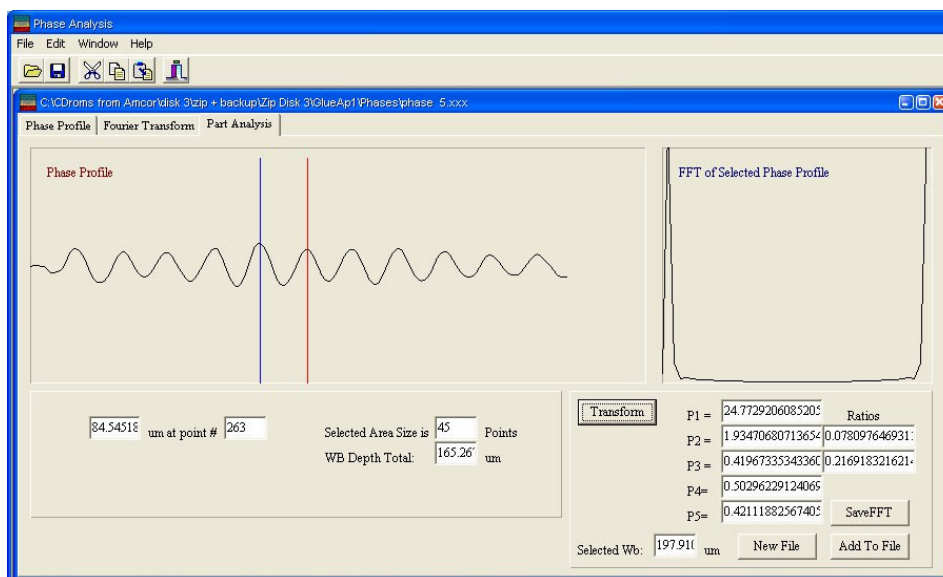


Figure [B.2] – Screen capture of shape and strain program showing washboarding profile, a single range selected (blue and red lines) for fourier transformation; and the fourier transformation of this range, on the right hand side of program window.

```

//-----
#ifndef MainH
#define MainH
//-----
#include "ChildWin.h"
#include <vcl\ComCtrls.hpp>
#include <vcl\ExtCtrls.hpp>
#include <vcl\Messages.hpp>
#include <vcl\Buttons.hpp>
#include <vcl\Dialogs.hpp>
#include <vcl\StdCtrls.hpp>
#include <vcl\Menus.hpp>
#include <vcl\Controls.hpp>
#include <vcl\Forms.hpp>
#include <vcl\Graphics.hpp>
#include <vcl\Classes.hpp>
#include <vcl\SysUtils.hpp>
#include <vcl\Windows.hpp>
#include <vcl\System.hpp>
//-----
class TMainForm : public TForm
{
__published:
    TMainMenu *MainMenu;
    TMenuItem *File1;
    TMenuItem *FileNewItem;
    TMenuItem *FileOpenItem;
    TMenuItem *FileCloseItem;
    TMenuItem *Window1;
    TMenuItem *Help1;
    TMenuItem *N1;
    TMenuItem *FileExitItem;
    TMenuItem *WindowCascadeItem;
    TMenuItem *WindowTileItem;
    TMenuItem *WindowArrangeItem;
    TMenuItem *HelpAboutItem;
    TOpenDialog *OpenDialog;
    TMenuItem *FileSaveItem;
    TMenuItem *FileSaveAsItem;
    TMenuItem *Edit1;
    TMenuItem *CutItem;
    TMenuItem *CopyItem;
    TMenuItem *PasteItem;
    TMenuItem *WindowMinimizeItem;
    TPanel *SpeedPanel;
    TSpeedButton *OpenBtn;
    TSpeedButton *SaveBtn;
    TSpeedButton *CutBtn;
    TSpeedButton *CopyBtn;
    TSpeedButton *PasteBtn;
    TSpeedButton *ExitBtn;
    TStatusBar *StatusBar;
    void __fastcall FormCreate(TObject *Sender);
    void __fastcall FileNewItemClick(TObject *Sender);
    void __fastcall WindowCascadeItemClick(TObject *Sender);
    void __fastcall UpdateMenuItems(TObject *Sender);
    void __fastcall WindowTileItemClick(TObject *Sender);
    void __fastcall WindowArrangeItemClick(TObject *Sender);
    void __fastcall FileCloseItemClick(TObject *Sender);
    void __fastcall FileOpenItemClick(TObject *Sender);
    void __fastcall FileExitItemClick(TObject *Sender);
    void __fastcall FileSaveItemClick(TObject *Sender);
    void __fastcall FileSaveAsItemClick(TObject *Sender);
    void __fastcall CutItemClick(TObject *Sender);
    void __fastcall CopyItemClick(TObject *Sender);
    void __fastcall PasteItemClick(TObject *Sender);
    void __fastcall WindowMinimizeItemClick(TObject *Sender);
    void __fastcall FormDestroy(TObject *Sender);

private:
    void __fastcall CreateMDIChild(const String Name);
    void __fastcall ShowHint(TObject *Sender);

public:
    virtual __fastcall TMainForm(TComponent *Owner);
};
//-----
extern TMainForm *MainForm;
extern TMDIChild *__fastcall MDIChildCreate(void);
//-----
#endif

```

```

#include <vcl.h>
#pragma hdrstop
#include "Main.h"

//-----
#pragma resource "*.dfm"
TMainForm *MainForm;
//-----
__fastcall TMainForm::TMainForm(TComponent *Owner)
    : TForm(Owner)
{
}
//-----
void __fastcall TMainForm::FormCreate(TObject *Sender)
{
    Application->OnHint = ShowHint;
    Screen->OnActiveFormChange = UpdateMenuItems;
}
//-----
void __fastcall TMainForm::ShowHint(TObject *Sender)
{
    StatusBar->SimpleText = Application->Hint;
}
//-----
void __fastcall TMainForm::CreateMDIChild(String Name)
{
    TMDIChild *Child;

    //--- create a new MDI child window ---
    Child = new TMDIChild(Application);
    Child->Caption = Name;

    Child->DrawBox(Name.c_str());
}

//-----
void __fastcall TMainForm::FileNewItemClick(TObject *Sender)
{
    CreateMDIChild("NONAME" + IntToStr(MDIChildCount + 1));
}
//-----
void __fastcall TMainForm::FileOpenItemClick(TObject *Sender)
{
    if (OpenDialog->Execute()){
        int i=0;
        while ( i < OpenDialog->Files->Count){
            // char tmp[10];
            // itoa (OpenDialog->Files->Count,tmp,10);
            //Caption = OpenDialog->Files->Strings[i];
            CreateMDIChild(OpenDialog->Files->Strings[i]);
            i++;
        }
    }
}
//-----
void __fastcall TMainForm::FileCloseItemClick(TObject *Sender)
{
    if (ActiveMDIChild)
        ActiveMDIChild->Close();
}
//-----
void __fastcall TMainForm::FileSaveItemClick(TObject *Sender)
{
    TMDIChild* sneaky ;
    sneaky = (TMDIChild*) ActiveMDIChild;
    sneaky -> SaveFFT();
    //---- save current file (ActiveMDIChild points to the window) ----
}
//-----
void __fastcall TMainForm::FileSaveAsItemClick(TObject *Sender)
{
    //---- save current file under new name ----
}
//-----
void __fastcall TMainForm::FileExitItemClick(TObject *Sender)
{
    Close();
}
//-----
void __fastcall TMainForm::CutItemClick(TObject *Sender)
{
    //---- cut selection to clipboard ----
}
//-----
void __fastcall TMainForm::CopyItemClick(TObject *Sender)
{
    //---- copy selection to clipboard ----
}
//-----
void __fastcall TMainForm::PasteItemClick(TObject *Sender)
{
    //---- paste from clipboard ----
}
//-----
void __fastcall TMainForm::WindowCascadeItemClick(TObject *Sender)
{
    Cascade();
}
//-----
void __fastcall TMainForm::WindowTileItemClick(TObject *Sender)
{
    Tile();
}
//-----

```

```

void __fastcall TMainForm::WindowArrangeItemClick(TObject *Sender)
{
    ArrangeIcons();
}
//-----
void __fastcall TMainForm::WindowMinimizeItemClick(TObject *Sender)
{
    int i;

    //---- Must be done backwards through the MDIChildren array ----
    for (i=MDIChildCount-1; i >= 0; i--)
        MDIChildren[i]->WindowState = wsMinimized;
}
//-----
void __fastcall TMainForm::UpdateMenuItems(TObject *Sender)
{
    FileCloseItem->Enabled = MDIChildCount > 0;
    FileSaveItem->Enabled = MDIChildCount > 0;
    FileSaveAsItem->Enabled = MDIChildCount > 0;
    CutItem->Enabled = MDIChildCount > 0;
    CopyItem->Enabled = MDIChildCount > 0;
    PasteItem->Enabled = MDIChildCount > 0;
    SaveBtn->Enabled = MDIChildCount > 0;
    CutBtn->Enabled = MDIChildCount > 0;
    CopyBtn->Enabled = MDIChildCount > 0;
    PasteBtn->Enabled = MDIChildCount > 0;
    WindowCascadeItem->Enabled = MDIChildCount > 0;
    WindowTileItem->Enabled = MDIChildCount > 0;
    WindowArrangeItem->Enabled = MDIChildCount > 0;
    WindowMinimizeItem->Enabled = MDIChildCount > 0;
}
//-----
void __fastcall TMainForm::FormDestroy(TObject *Sender)
{
    Screen->OnActiveFormChange = NULL;
}
//-----

```

```

//-----
#ifndef ChildWinH
#define ChildWinH
//-----
#include <vcl\Controls.hpp>
#include <vcl\Forms.hpp>
#include <vcl\Graphics.hpp>
#include <vcl\Classes.hpp>
#include <vcl\Windows.hpp>
#include <vcl\System.hpp>
#include <vcl\ExtCtrls.hpp>
#include "drawarrayonpaintbox.h"
#include <vcl\StdCtrls.hpp>
#include <vcl\ComCtrls.hpp>
#include <vcl\Dialogs.hpp>
//-----
class TMDIChild : public TForm
{
__published:
    TPageControl *PageControl1;
    TTabSheet *TabSheet1;
    TTabSheet *TabSheet2;
    TPaintBox *PaintBox1;
    TPaintBox *RealFFTPB;
    TPaintBox *ImagFFTPB;
    TPaintBox *MagFFTPB;
    TLabel *Label1;
    TLabel *Label2;
    TLabel *Label3;
    TLabel *Label4;
    TLabel *Label5;
    TLabel *Label6;
    TLabel *Label7;
    TEdit *YValue;
    TEdit *YDifference;
    TEdit *DeltaLEdit;
    TEdit *AftLengthEdit;
    TEdit *PreLengthEdit;
    TEdit *StrainEdit;
    TTrackBar *ZoomBar;
    TBevel *Bevel1;
    TBevel *Bevel2;
    TBevel *Bevel3;
    TSaveDialog *SaveDialog1;
    TEdit *P1Edit;
    TEdit *P2Edit;
    TLabel *P1;
    TLabel *Label8;
    TLabel *Label9;
    TEdit *RatioEdit;
    TLabel *Label10;
    TEdit *MaxAtEdit;
    TEdit *P2AtEdit;
    TLabel *label;
    TLabel *Label11;
    TTabSheet *TabSheet3;
    TPaintBox *SelectFFTPhase;
    TPaintBox *FFTTransform;
    TEdit *HeightText;
    TLabel *labeum;
    TEdit *PartPointText;
    TLabel *Label12;
    TEdit *SAreaText;
    TLabel *Label13;
    TButton *Transform;
    TLabel *Label14;
    TLabel *Label15;
    TLabel *Label16;
    TEdit *PartP1Text;
    TEdit *PartP2Text;
    TEdit *PartP3Text;
    TBevel *Bevel4;
    TBevel *Bevel5;
    TEdit *R1Text;
    TLabel *Label17;
    TEdit *R2Text;
    TButton *AddToFileButton;
    TButton *NewFileButton;
    TEdit *SDText;
    TLabel *Label18;
    TEdit *PartSDText;
    TBevel *Bevel6;
    TBevel *Bevel7;
    TLabel *Label19;
    TStaticText *StaticText1;
    TStaticText *StaticText2;
    TStaticText *StaticText3;
    TStaticText *StaticText4;
    TSaveDialog *SaveDialog2;
    TBevel *Bevel8;
    TBevel *Bevel9;
    TEdit *PartP4Text;
    TEdit *PartP5Text;
    TStaticText *StaticText5;
    TStaticText *StaticText6;
    TSaveDialog *SaveDialog3;
    TButton *SaveFFTButton;
    void __fastcall FormClose(TObject *Sender, TCloseAction &Action);

    void __fastcall FormShow(TObject *Sender);
    void __fastcall PaintBox1MouseDown(TObject *Sender, TMouseButton Button,
    TShiftState Shift, int X, int Y);
    void __fastcall PaintBox1MouseMove(TObject *Sender, TShiftState Shift, int X,
    int Y);

    void __fastcall RealFFTPBPaint(TObject *Sender);
    void __fastcall ImagFFTPBPaint(TObject *Sender);

```

```

void __fastcall MagFFTPBPaint(TObject *Sender);
void __fastcall ZoomBarChange(TObject *Sender);
void __fastcall Button1Click(TObject *Sender);

void __fastcall TransformClick(TObject *Sender);
void __fastcall SelectFFTPhaseMouseDown(TObject *Sender,
    TMouseButton Button, TShiftState Shift, int X, int Y);
void __fastcall SelectFFTPhaseMouseUp(TObject *Sender,
    TMouseButton Button, TShiftState Shift, int X, int Y);
void __fastcall SelectFFTPhaseMouseMove(TObject *Sender,
    TShiftState Shift, int X, int Y);
void __fastcall NewFileButtonClick(TObject *Sender);
void __fastcall AddToFileButtonClick(TObject *Sender);

void __fastcall SaveFFTButtonClick(TObject *Sender);
private:
    drawarrayonbox* putit;
    int sel1, sel2;
    bool SelectStart;
    bool DataFileActive;

public:
    float ZoomFactor;
    virtual __fastcall TMDIChild(TComponent *Owner);
    void __fastcall DrawBox(char* file);
    void __fastcall UpdateBox(void);
    void SaveFFT(void);

};
//-----
#endif

```

```

//-----
#include <vc1.h>
#pragma hdrstop

#include "ChildWin.h"
//-----
#pragma resource "*.dfm"
//-----
__fastcall TMDIChild::TMDIChild(TComponent *Owner)
    : TForm(Owner)
{
    ZoomFactor = 1;
    SelectStart = false;
    DataFileActive = false;
}
void TMDIChild::SaveFFT(void)
{
    if (SaveDialog1->Execute()){
        putit->SaveFFT(SaveDialog1->FileName.c_str());
    }
}

//-----
void __fastcall TMDIChild::FormClose(TObject *Sender, TCloseAction &Action)
{
    delete putit;
    Action = caFree;
}
//-----

void __fastcall TMDIChild::DrawBox(char* file)
{
    putit = new
    drawarrayonbox(PreLengthEdit, AftLengthEdit, DeltaLEdit, StrainEdit, PaintBox1, RealFFTPB, ImagFFTPB, MagFFTPB, file);
    putit->DoIt();
    P1Edit->Text = putit->P1;
    P2Edit->Text = putit->P2;
    MaxAtEdit->Text = putit->MaxIsAt;
    P2AtEdit->Text = putit->P2IsAt;
    RatioEdit->Text = putit->P2 / putit->P1;
}

void __fastcall TMDIChild::UpdateBox(void)
{
    putit->DrawArray();
}

void __fastcall TMDIChild::FormShow(TObject *Sender)
{
    putit->DrawArray();
    putit->DrawArrayOnPartPage(SelectFFTPhase);
    SDText -> Text = putit -> GetWhole3SD();
}
//-----

void __fastcall TMDIChild::PaintBox1MouseDown(TObject *Sender,
    TMouseButton Button, TShiftState Shift, int X, int Y)
{
    putit->SetPoint(X);
    switch(Button){
        case mbLeft:
            putit -> low = putit -> GetValueAtPoint();
            break;
        case mbRight:
            putit -> high = putit -> GetValueAtPoint();
            break;
    }

    YDifference->Text = putit->high - putit->low;
}
//-----
void __fastcall TMDIChild::PaintBox1MouseMove(TObject *Sender,
    TShiftState Shift, int X, int Y)
{
    putit->SetPoint(X);
    YValue->Text = putit -> GetValueAtPoint();
}
//-----
void __fastcall TMDIChild::RealFFTPBPaint(TObject *Sender)
{
    putit -> DrawRealFFT(ZoomFactor);
}
//-----
void __fastcall TMDIChild::ImagFFTPBPaint(TObject *Sender)
{
    putit -> DrawImagFFT(ZoomFactor);
}
//-----
void __fastcall TMDIChild::MagFFTPBPaint(TObject *Sender)
{
    putit -> DrawMagFFT(ZoomFactor);
}
//-----
void __fastcall TMDIChild::ZoomBarChange(TObject *Sender)
{
    ZoomFactor = ZoomBar -> Position * .05 ;
    RealFFTPB-> Hide();
    putit -> DrawRealFFT(ZoomFactor);
    RealFFTPB-> Show();
    ImagFFTPB-> Hide();
    putit -> DrawImagFFT(ZoomFactor);
    ImagFFTPB-> Show();
    MagFFTPB-> Hide();
    putit -> DrawMagFFT(ZoomFactor);
}

```



```

        MagFFTPB-> Show();
    }
    //-----
    void __fastcall TMDIChild::Button1Click(TObject *Sender)
    {
        putit -> DoIFFT();
    }
    //-----

    void __fastcall TMDIChild::TransformClick(TObject *Sender)
    {
        putit ->
        PartTransform(sel1, sel2, FFTTransform, PartP1Text, PartP2Text, PartP3Text, PartP4Text, PartP5Text, R1Text, R2Text, PartSDText);
    }
    //-----

    void __fastcall TMDIChild::SelectFFTPhaseMouseDown(TObject *Sender,
        TMouseButton Button, TShiftState Shift, int X, int Y)
    {
        if (SelectStart == false ){
            putit -> DrawArrayOnPartPage(SelectFFTPhase);
            SelectStart = true;
        }

        SelectFFTPhase -> Canvas ->Pen-> Color = TColor(clBlue);
        SelectFFTPhase -> Canvas ->MoveTo(X, SelectFFTPhase->Top);
        SelectFFTPhase -> Canvas ->LineTo(X, SelectFFTPhase->Top+SelectFFTPhase->Height);
        SelectFFTPhase -> Canvas ->Pen-> Color = TColor(clBlack);
        sel1=X;
    }
    //-----

    void __fastcall TMDIChild::SelectFFTPhaseMouseUp(TObject *Sender,
        TMouseButton Button, TShiftState Shift, int X, int Y)
    {
        sel2=X;
        PartPointText->Text = X;
        SAreaText->Text = X-sel1;
        SelectStart = false;

        SelectFFTPhase -> Canvas ->Pen-> Color = TColor(clRed);
        SelectFFTPhase -> Canvas ->MoveTo(X, SelectFFTPhase->Top);
        SelectFFTPhase -> Canvas ->LineTo(X, SelectFFTPhase->Top+SelectFFTPhase->Height);
        SelectFFTPhase -> Canvas ->Pen-> Color = TColor(clBlack);
    }
    //-----

    void __fastcall TMDIChild::SelectFFTPhaseMouseMove(TObject *Sender,
        TShiftState Shift, int X, int Y)
    {
        if(Shift.Contains(ssLeft)){
            HeightText->Text = putit->HeightAtPartMouse(X);
            PartPointText->Text = X;
            SAreaText->Text = X-sel1;
        }
    }
    //-----

    void __fastcall TMDIChild::NewFileButtonClick(TObject *Sender)
    {
        if (putit->MakeNewDataFile(SaveDialog2)==true){
            DataFileActive = true;
        }
    }
    //-----

    void __fastcall TMDIChild::AddToFileButtonClick(TObject *Sender)
    {
        if(DataFileActive==false){
            if (putit->MakeNewDataFile(SaveDialog2)==true){
                DataFileActive = true;
                putit-> AddToDataFile();
            }
        }else{
            putit-> AddToDataFile();
        }
    }
    //-----

    void __fastcall TMDIChild::SaveFFTButtonClick(TObject *Sender)
    {
        putit -> SavePartFFT(SaveDialog3);
    }
    //-----

```

```

//-----
#ifndef drawarrayonpaintboxH
#define drawarrayonpaintboxH
#include <stdio.h>

extern "C" int fft(float*,float*, long ,long ,long ,int );

class drawarrayonbox{
private:
    void LoadArray(void);
    void FindFFTPeaks(void);
    char* filename;
    TPaintBox* PB;
    TPaintBox *RealFFTPB;
    TPaintBox *ImagFFTPB;
    TPaintBox *MagFFTPB;

    int nlines;
    float* array;
    float* xarray;
    float* imag;
    float* RealFFTArray;
    float* ImagFFTArray;
    float* MagFFTArray;
    int SelPoint;
    TEdit* PreLengthEdit;
    TEdit* AftLengthEdit;
    TEdit* DeltaEdit;
    TEdit* StrainEdit;

    FILE* DataFile;
    bool DataFileActive;
    float W_3sd, W_P1, W_P2, W_Ratio, P_3sd, P_P1, P_P2, P_P3,P_P4, P_P5, P_R1, P_R2;
    float Real[512],Imag[512],Length;
public:
    float P1, P2;
    int P2IsAt,MaxIsAt;
    float low,high;

    void SetPoint(int X);
    float GetValueAtPoint(void);

    drawarrayonbox(
        TEdit* PLEdit,
        TEdit* ALEdit,
        TEdit* DLEdit,
        TEdit* SEdit,
        TPaintBox* paintbox,TPaintBox* paintbox2, TPaintBox* paintbox3, TPaintBox* paintbox4,
        char* file);
    void DrawArray(void);
    void DrawArrayOnPartPage(TPaintBox* PB);
    void CalculateStrain(void);
    void DoIt(void);
    void DoFFT(void);
    void DoIFFT(void);
    void DrawRealFFT(float zoom);
    void DrawImagFFT(float zoom);
    void DrawMagFFT(float zoom);
    void SaveFFT(char* filename);
    float HeightAtPartMouse(int place);
    void PartTransform(int Sel1,int Sel2, TPaintBox* PB, TEdit* E1, TEdit* E2, TEdit* E3,TEdit* E4,TEdit* E5,TEdit* R1,
TEdit* R2, TEdit* SDText);
    bool MakeNewDataFile(TSaveDialog* SaveDialog2);
    void AddToDataFile(void);
    float GetWhole3SD(void);
    float GetSD(float* Array, int Length);
    void SavePartFFT(TSaveDialog* SaveDialog2);
    ~drawarrayonbox();
};
//-----
#endif

```

```

//-----
#include <vc1\vc1.h>
#include <stdio.h>
#include <math.h>
#pragma hdrstop
#include "drawarrayonpaintbox.h"

drawarrayonbox::drawarrayonbox(
    TEdit* PLEdit,
    TEdit* ALEdit,
    TEdit* DLEdit,
    TEdit* SEdit,
    TPaintBox* paintbox, TPaintBox* paintbox2, TPaintBox* paintbox3, TPaintBox* paintbox4,
    char* file)
{
    PreLengthEdit = PLEdit;
    AftLengthEdit = ALEdit;
    DeltaLEdit = DLEdit;
    StrainEdit = SEdit;
    PB = paintbox;
    RealFFTPB = paintbox2;
    ImagFFTPB = paintbox3;
    MagFFTPB = paintbox4;
    filename = file;
    high=low=0;
    SelPoint = 0;
    DataFileActive = false;
}

void drawarrayonbox::SaveFFT(char* filename)
{
    FILE* out;
    if ((out = fopen(filename, "wt"))== NULL){
        return ;
    }
    for (int i = 0 ; i< nlines; i++){
        fprintf(out,"%f %f %f\n", RealFFTArray[i], ImagFFTArray[i], MagFFTArray[i]);
    }
    fclose(out);
}

void drawarrayonbox::SavePartFFT(TSaveDialog* SaveDialog3)
{
    FILE* out;
    if (SaveDialog3->Execute()){
        if ((out = fopen(SaveDialog3->FileName.c_str(), "wt"))== NULL){
            return;
        }else{
            for (int x = 0 ; x< Length; x++){
                float r = Real[x];

                fprintf(out,"%f\n", r);
            }
        }
    }
    fclose(out);
}

void drawarrayonbox::LoadArray(void)
{
    FILE* in;
    float* nullarray;
    nlines =0;
    if ((in = fopen(filename, "rt"))== NULL){
        return ;
    }
    for(; !feof(in);){
        if(getc(in) == '\n') nlines++;
    }
    array = new float[nlines+1];
    imag = new float[nlines+1];
    xarray = new float[nlines+1];
    RealFFTArray = new float[nlines+1];
    ImagFFTArray = new float[nlines+1];
    MagFFTArray = new float[nlines+1];

    rewind(in);
    for(int i=0; i<nlines; i++){
        fscanf(in,"%f %f\n",&xarray[i],&array[i]);
        array[i] *= 168.7;
        RealFFTArray[i] = array[i];
        ImagFFTArray[i] = 0;
    }
}

drawarrayonbox::~drawarrayonbox(){
    if (DataFileActive == true) fclose(DataFile);
    delete[] array;
    delete[] imag;
    delete[] xarray;
    delete[] RealFFTArray;
    delete[] ImagFFTArray;
    delete[] MagFFTArray;
}

float drawarrayonbox::GetValueAtPoint (void)
{
    return array[SelPoint];
}

void drawarrayonbox::SetPoint (int X)
{
    SelPoint = X;
}

void drawarrayonbox::DrawArrayOnPartPage (TPaintBox* PB)

```

```

{
    int cw, ch;

    cw = PB->ClientWidth;
    ch = PB->ClientHeight;

    float max = 600, min = -600;

    float m= ch/(max-min);
    float c= max;

    PB->Canvas->Pen->Color = TColor(clBlack);
    PB->Canvas->MoveTo(0, (c-array[0])*m);
    for(int x = 0; x < nlines; x++){
        PB->Canvas->LineTo(x, (c-array[x])*m);
    }
}

void drawarrayonbox::DrawArray(void)
{
    int cw, ch;

    cw = PB->ClientWidth;
    ch = PB->ClientHeight;

    float max = 600, min = -600;

    float m= ch/(max-min);
    float c= max;

    PB->Canvas->Pen->Color = TColor(clBlack);
    PB->Canvas->MoveTo(0, (c-array[0])*m);
    for(int x = 0; x < nlines; x++){
        PB->Canvas->LineTo(x, (c-array[x])*m);
    }
}

void drawarrayonbox::CalculateStrain(void)
{
    float PreLength=0, AftLength=0, DeltaL=0,DeltaX,Strain=0;
    int i,x;

    DeltaX = xarray[1]*1000; // um
    PreLength = DeltaX*(nlines); // um
    for (x = 0; x < nlines; x++){ // away from filter startup
        AftLength += sqrt((array[x]*array[x])+(DeltaX*DeltaX));
    }
    DeltaL= AftLength-PreLength;
    Strain= DeltaL/PreLength;

    PreLengthEdit -> Text = PreLength /1000.0; // mm
    AftLengthEdit -> Text = AftLength /1000.0; //mm
    DeltaLEdit -> Text = DeltaL; // um
    StrainEdit -> Text = Strain;
}

void drawarrayonbox::DrawRealFFT(float zoom)
{
    int cw, ch;

    cw = RealFFTPB->ClientWidth;
    ch = RealFFTPB->ClientHeight;

    float max = 200, min = -200;

    float m= ch/(max-min);
    float c= max;

    RealFFTPB->Canvas->Pen->Color = TColor(clBlack);
    RealFFTPB->Canvas->MoveTo(0, (c-RealFFTArray[0]*zoom)*m);
    for(int x = 0; x < nlines; x++){
        RealFFTPB->Canvas->LineTo(x, (c-RealFFTArray[x]*zoom)*m);
    }
}

void drawarrayonbox::DrawImagFFT(float zoom)
{
    int cw, ch;

    cw = ImagFFTPB->ClientWidth;
    ch = ImagFFTPB->ClientHeight;

    float max = 200, min = -200;

    float m= ch/(max-min);
    float c= max;

    ImagFFTPB->Canvas->Pen->Color = TColor(clBlack);
    ImagFFTPB->Canvas->MoveTo(0, (c-ImagFFTArray[0]*zoom)*m);
    for(int x = 0; x < nlines; x++){
        ImagFFTPB->Canvas->LineTo(x, (c-ImagFFTArray[x]*zoom)*m);
    }
}

void drawarrayonbox::DrawMagFFT(float zoom)
{
    int cw, ch;

    cw = MagFFTPB->ClientWidth;
    ch = MagFFTPB->ClientHeight;

    float max = 200, min = 0;

    float m= ch/(max-min);
    float c= max;

    MagFFTPB->Canvas->Pen->Color = TColor(clBlack);
    MagFFTPB->Canvas->MoveTo(0, (c-MagFFTArray[0]*zoom)*m);
    for(int x = 0; x < nlines; x++){
        MagFFTPB->Canvas->LineTo(x, (c-MagFFTArray[x]*zoom)*m);
    }
}

```

```

    }
    FindFFTPeaks();
}
void drawarrayonbox::FindFFTPeaks(void)
{
    float BigNumber, MagNumber;
    int i;
    for(i=BigNumber = 0; i< nlines/4; i++){
        MagNumber = MagFFTArray[i];
        if( BigNumber < MagNumber){
            BigNumber = MagNumber;
            MaxIsAt = i;
        }
    }
    W_P1 = P1 = (float) BigNumber;
    for(i=MaxIsAt + 6, BigNumber = 0; i< nlines/4; i++){
        MagNumber = MagFFTArray[i];
        if( BigNumber < MagNumber){
            BigNumber = MagNumber;
            P2IsAt = i;
        }
    }
    W_P2 = P2 = (float) BigNumber;
    W_Ratio = P2/P1;
}

void drawarrayonbox::DoFFT (void){
    fft (RealFFTArray, ImagFFTArray, nlines+1, nlines+1, nlines+1, 1);
    for (int x= 0; x< nlines; x++){
        float r= RealFFTArray[x], i = ImagFFTArray[x];
        MagFFTArray[x] = sqrt((i*i)+(r*r))/(nlines+1*2.0);
        RealFFTArray[x] /= (nlines+1*2.0);
        ImagFFTArray[x] /= (nlines+1*2.0);
    }
}

void drawarrayonbox::PartTransform(int Sel1, int Sel2, TPaintBox* PB, TEdit* E1, TEdit* E2, TEdit* E3, TEdit* E4, TEdit* E5,
TEdit* R1, TEdit* R2, TEdit* SDText)
{
    int i, j;
    Length = Sel2-Sel1;
    for(i = Sel1, j = 0; j < Length; j++, i++){
        Real[j] = array[i];
        Imag[j] = 0;
    }

    SDText->Text = P_3sd = 3 * GetSD(Real, Length);

    Length = fft(Real, Imag, Length, Length, Length, 1);
    for (int x= 0; x< Length; x++){
        float r= Real[x], i = Imag[x];
        Real[x] = sqrt((i*i)+(r*r))/(2*Length);
    }
    E1->Text = P_P1= Real[1];
    E2->Text = P_P2= Real[2];
    E3->Text = P_P3= Real[3];
    E4->Text = P_P4= Real[4];
    E5->Text = P_P5= Real[5];

    R1->Text = P_R1 = P_P2/P_P1;
    R2->Text = P_R2 = P_P3/P_P2;

    float zoom = 10;
    int cw, ch;
    cw = PB->ClientWidth;
    ch = PB->ClientHeight;

    PB->Canvas->Pen->Color = TColor(clBlack);
    PB->Canvas->MoveTo(0, (ch-Real[0]*zoom));
    for(int x = 0; x< Length; x++){
        PB->Canvas->LineTo((x*cw)/Length, (ch-Real[x]*zoom));
    }
}

void drawarrayonbox::DoIFFT (void){
    for (int x = 0 ; x < nlines; x++){
        array[x] = RealFFTArray[x]/(nlines);
        imag[x] = ImagFFTArray[x]/(nlines);
    }
    array[0] = imag[0] = 0;
    for (int x= 1; x< 3; x++){
        array[x] = imag[x] = array[nlines-x] = imag[nlines-x] = 0;
    }
    for (int x= 50; x< 255; x++){
        array[x] = imag[x] = array[nlines-1-x] = imag[nlines-1-x] = 0;
    }
    fft (array, imag, nlines+1, nlines+1, nlines+1, -1);
    DrawArray();
    for(int i=0; i<nlines; i++){
        RealFFTArray[i] = array[i] ;
        ImagFFTArray[i] = 0;
    }
    DoFFT();
    DrawRealFFT(1);
    DrawImagFFT(1);
    DrawMagFFT(1);
}

float drawarrayonbox::HeightAtPartMouse(int place)
{
    return array[place];
}

bool drawarrayonbox::MakeNewDataFile(TSaveDialog* SaveDialog2)

```

```

{
    if (SaveDialog2->Execute()){
        if ((DataFile = fopen(SaveDialog2->FileName.c_str(), "a+t"))== NULL){
            return false;
        }else{
            DataFileActive = true;
        }
    }

    fprintf(DataFile,"W_3sd, W_P1, W_P2, W_Ratio, P_3sd, P_P1, P_P2, P_P3, P_R1, P_R2\n");
    return true;
}
void drawarrayonbox::AddToDataFile(void)
{
    if(DataFileActive == true){
        fprintf(DataFile,"%f,%f,%f,%f,%f,%f,%f,%f,%f,%f\n",W_3sd, W_P1, W_P2, W_Ratio, P_3sd, P_P1, P_P2, P_P3, P_P4,
P_P5,
                P_R1, P_R2);
    }
}
float drawarrayonbox::GetWhole3SD(void)
{
    return (W_3sd=3* GetSD(array,nlines));
}

float drawarrayonbox::GetSD(float* Array,int Length)
{
    int x;
    float mean=0, Sx2=0, SD;
    for(x = 2; x < Length; x++){
        Sx2 += Array[x]*Array[x];
        mean += Array[x];
    }
    if (Length == 0) Length =1; // to stop crashing
    mean /= (float)Length;

    SD = sqrt(Sx2/Length - mean*mean);

    return SD;
    return 0;
}

void drawarrayonbox::DoIt(void)
{
    LoadArray();
    DrawArray();
    DoFFT();
    DrawRealFFT(10);
    DrawImagFFT(10);
    DrawMagFFT(10);
    CalculateStrain();
}
//-----

```

Appendix C

Paper Properties used for Finite Element Analysis (FEA)

Virgin Fibre KLB21 Manufactured at Maryvale machine #4							
Relative humidity (%)	Moisture (%)	Grammage (gsm)	Thickness (um)	Stiffness MD (kN/m)	Young's Modulus MD (MPa)	Stiffness CD (kN/m)	Young's Modulus CD (MPa)
23	5.3	197	304	2146	7059	742	2441
50	8.1	203	309	1993	6450	663	2146
63	11	207	317	1839	5801	603	1902
78	12.4	209	320	1661	5191	522	1631
90	15.8	216	329	1338	4067	398	1210