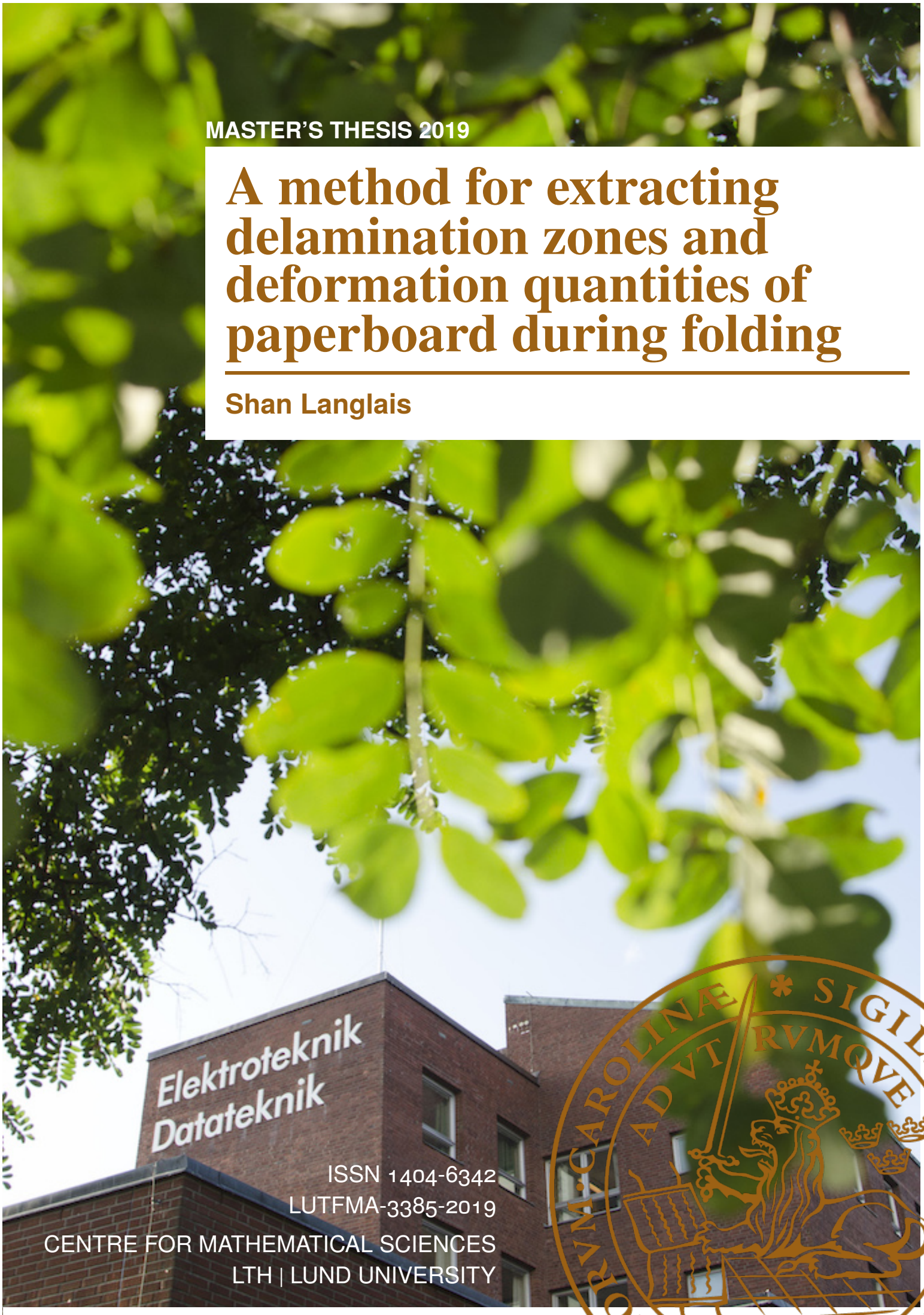

MASTER'S THESIS 2019

A method for extracting delamination zones and deformation quantities of paperboard during folding

Shan Langlais

ISSN 1404-6342
LUTFMA-3385-2019

CENTRE FOR MATHEMATICAL SCIENCES
LTH | LUND UNIVERSITY

EXAMENSARBETE
Matematikcentrum

A method for extracting delamination zones and deformation quantities of paperboard during folding

Shan Langlais

A method for extracting delamination zones and deformation quantities of paperboard during folding

Shan Langlais
dat13sla@student.lu.se

August 28, 2019

Master's thesis work carried out at Tetra Pak Packaging Solutions AB.

Supervisors: Magnus Oskarsson, magnuso@maths.lth.se
Eric Borgqvist, eric.borgqvist@tetrapak.com
Johan Tryding, johan.tryding@tetrapak.com

Examiner: Kalle Åström, kalle@maths.lth.se

Abstract

This study has been conducted at Tetra Pak to study deformation in liquid packaging paperboard during the folding process.

When a package is formed in the filling machines, the paperboard is folded according to the crease lines. It is of great interest to study the folding process since it is crucial to the final quality of the package. Mathematical models are used for simulation of paperboard but they require real world measurements and data for calibration and verification.

A method utilizing the optical flow in video sequences of the folding process of paperboard, which captures the geometrical changes that the material undergoes during folding, has been developed. Fields such as displacement, deformation gradient, strain tensor, the Jacobian of the deformation, and others, can be obtained for actual paperboard samples. Furthermore, delamination zones and openings in the paperboard can be obtained automatically. All metrics and quantities can be given as a function of time, or folding angle, allowing for study of the translation and propagation of these.

The method has been verified by applying it on rendered simulations of the folding process, of which ground truth data is available.

Quantitative and qualitative analyses have been performed on video sequences of the folding process of paperboard. The most relevant quantities, fields, and metrics have been extracted for all samples. An analysis of delamination zone area as a function of folding angle has also been conducted.

Keywords: liquid paperboard package, continuum mechanics, optical flow, deformation gradient, delamination, folding

Acknowledgements

I would like to thank Eric Borgqvist and Johan Tryding at Tetra Pak for their exceptional guidance and support in this interesting and challenging field. Also a big thanks to Erik Bergvall at Tetra Pak for his interest in this work and for all his good ideas. I would also like to thank Magnus Oskarsson at The Centre for Mathematical Sciences for his help during the project and for his feedback on this report, and Flavius Gruian at The Department of Computer Science for the excellent LaTeXtemplate.

Contents

1 Introduction **7**

2 Background **9**

2.1 Paperboard as a packaging material 9
2.2 Folding and creasing of paperboard 10
2.3 Simulating paperboard behaviour 11
2.4 Continuum mechanics 12
2.4.1 Undeformed and deformed configurations 12
2.4.2 Lagrangian and Eulerian descriptions 12
2.4.3 Useful tensors 12
2.5 Shape functions in the finite element method 13
2.6 Optical flow 14
2.6.1 Optical flow approximation methods 16

3 Method **19**

3.1 Overview 19
3.2 Theory 21
3.2.1 Optical flow reconstruction 21
3.2.2 Tracking of material points 21
3.2.3 Extraction of the deformation gradient 23
3.2.4 Detection of structural differences 24
3.3 Implementation 25
3.3.1 Optical flow extraction 25
3.3.2 Detection and extraction of structural differences 26
3.3.3 Application of optical flow and difference fields 27
3.3.4 Deformation gradient computation 28
3.3.5 Ground truth extraction and computation 28
3.3.6 Rendering of simulation 28

4 Results **31**
4.1 Ground truth comparisons . . . 32
4.2 Delamination . . . 32
4.3 Delamination analysis . . . 32

5 Discussion **43**
5.1 Method verification . . . 43
5.1.1 Considerations . . . 43
5.1.2 Optical flow tracking . . . 44
5.1.3 Ground truth comparisons . . . 44
5.1.4 Summary of method verification . . . 45
5.2 Delamination and deformation fields of actual samples . . . 45
5.2.1 Optical flow tracking . . . 45
5.2.2 Differences for estimating delamination . . . 45
5.2.3 Delamination zones . . . 46
5.2.4 Deformation fields . . . 47
5.2.5 Statistical analysis . . . 47
5.3 Future work . . . 48

Bibliography **49**

Appendix A Dataset **55**

Chapter 1
Introduction

Liquid paperboard packages are typically made of a laminate of different materials such as paperboard, polymer, and aluminum. The paperboard is the material making up most of the volume and stiffness of the laminate and it thus has the highest influence on the structural integrity of the package. The paperboard usually consists of many different layers, plies, which exhibit different characteristics.

Forming a package involves folding of the laminate, in which a certain amount of delamination of the different layers of the paperboard is desired. This allows for folding such that the risk for wrinkles and cracks is minimized. Characterisation of the delamination process thus allows for evaluating the quality of a fold.

In the packaging industry, certain constraints on different aspects of the delamination process when folding are imposed on the pulp and paper suppliers. Not all constraints can however be quantitatively motivated since one usually studies delamination zones qualitatively, by hand, which makes it tedious to gather enough data to be able to draw statistical conclusions about the process.

Accurate folding simulations are, in part for the reason above, highly sought after since a good quality fold when forming a package is of critical importance both aesthetically and structurally. Wrinkles that impact the aesthetics of the formed package negatively and cracks that destroy the aseptic capability of the liquid package are often the result of bad folds. It is therefore of great importance to be able to predict how different paperboards behave during these processes.

Mathematical models can be used for simulation of many situations but since paperboard is a complicated, highly anisotropic material, these models require parameters which are either hard to interpret physically or which are difficult to measure in reality. Additionally, these models might be missing certain physical mechanisms and processes and they thus need to be verified and calibrated using real world measurements and data.

Measurements which correlate well to certain aspects of a fold can be obtained but they are not representative of the whole process. It is often also difficult to know what is actually measured and to isolate effects of the parameter or parameters of interest.

There are two main points motivating the work described in this report. The first is that automatic analysis of delamination zones allows for quantitative analysis of these processes which in turn help reveal facts about the folding process but also allow for evaluation of different paperboards from different suppliers. Quantities such as total delamination area at a specific folding angle, rate of change of delamination zone area, delamination position, angle at which delamination first occurs, and so on, are all of interest.

The second point is that it is desirable to capture and quantify the geometrical changes that a paperboard undergoes during folding. This allows for direct comparison of actual folds with simulations; metrics and quantities such as strain fields and deformation gradients can be measured and compared in each time step. Thus it is a way to verify that simulations predict what actually happens empirically.

There are current methods employing digital image correlation to extract strain fields of real specimens. Most of these methods require spraying a speckle pattern onto the material so as to be able to track each speckle and the area surrounding it. This, however, contaminates the specimen and might also influence the mechanical response of the material. For out of plane warping it is possible to do digital image correlation without physically spraying the specimen. These speckle free methods often require complicated camera setups, such as using a projector in conjunction with other cameras for applying virtual speckles [1].

Chapter 2
Background

2.1 Paperboard as a packaging material

One of the reasons for using paperboard as the main material in liquid packages stems from the fact that it is highly anisotropic. It exhibits different behaviour in different directions. This makes it suitable for packaging applications, since it can be strong in certain directions and weak in other directions, which is a characteristic providing for good converting properties such as foldability and creasability, but also for end use capabilities [2]. In fact, paperboard is often regarded to be an orthotropic material since the mechanical properties differ in the mutually orthogonal directions, MD, CD, and ZD, illustrated in figure 2.1.

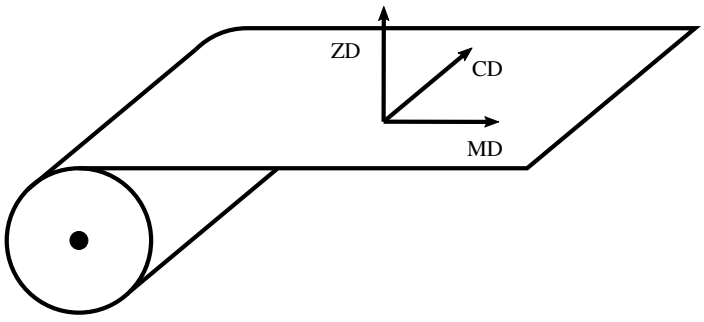

Figure 2.1: The three principal directions MD (machine direction), CD (cross direction), and ZD (out-of-plane direction).

The difference in directions exists due to the fact that the paperboard is made up of fibers. When paperboard is manufactured, a wet mass of fibers is sprayed onto a moving web and the alignment of these is a function of the speed at which the web moves. Because of the fiber alignment, the material is 2-3 times more stiff in MD than in CD, and 100 times more stiff than in ZD [3].

2.2 Folding and creasing of paperboard

When forming a package, the laminate containing the paperboard needs to be folded. However, doing this on an undeformed paperboard as it is results in an uneven fold with wrinkles and possibly cracks. To alleviate this issue one must first purposefully damage the board in a process called creasing in which a male die pushes the paperboard into a female die, along a crease line. Figure 2.2 illustrates the creasing procedure. The depth of creasing is instrumental in providing a good fold. If the crease depth is too large the paperboard might crack during the creasing. If it is too small the paperboard might not have been deformed enough to allow for the creasing operation to improve the quality of a fold [4].

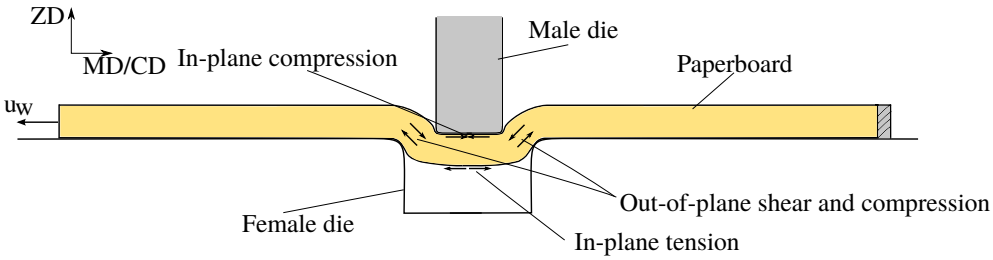

Figure 2.2: Creasing of paperboard [5]

The creased paperboard has been subject to shear stress between the different layers of the board, and between the fibers in the ZD, out-of-plane direction. Delamination of a paperboard is thus promoted, if it hasn't already occured in the creasing process. Additionally, by creating "hinges" where the paperboard comes into contact with the upper edges of the female die, folding offers less resistance to bending. This reduces the energy put into the system during a fold, decreasing the risk for cracks.

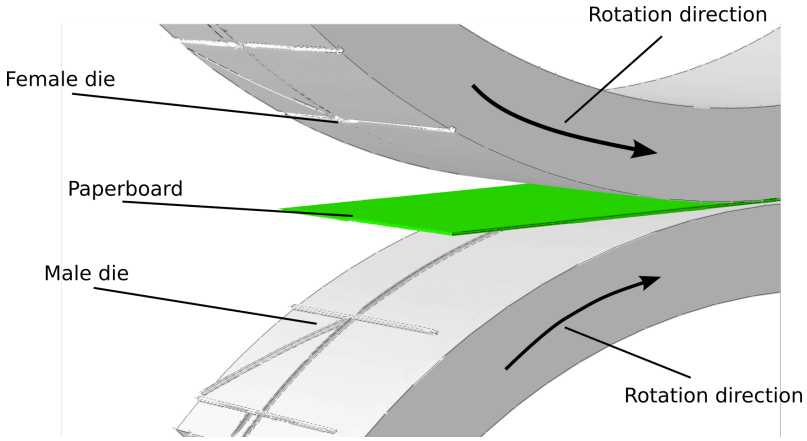

Figure 2.3: Rotation creasing of paperboard [5]

By folding the paperboard so that the bulge created by pushing the paperboard into the female die is on the inside of a fold, the paperboard may grow in ZD direction, has the paperboard been creased to allow for sufficient delamination. This allows for the outer layer (on the opposite side of the bulge) to be subject to less stress since the strain will be smaller than if the outer layer had to form around the material on the inside of the fold, which is illustrated in figure 2.4. A crease depth which is too small will thus not allow

the inner layers to make room for the fold, which may lead to the outer layer cracking [6]. A sufficient amount of delamination when folding is thus of high importance for creating folds which exhibit desired properties such as ease of folding and lack of cracks and wrinkles.

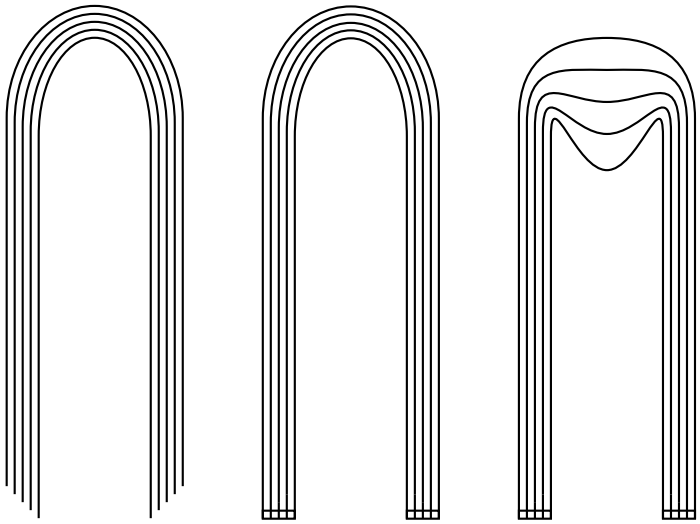

Figure 2.4: *Left*: folding of laminate without fastening the layers at the tips which allows for the layers to slide, preventing excessive stress at the tip. *Middle*: Fastening the tips and folding without delamination. Large stress at the tip with risk for cracking. *Right*: Fastening the tips and folding with delamination which alleviates the stress of the outer layers.

2.3 Simulating paperboard behaviour

Mathematical models for efficiently simulating paperboard behaviour are used with great success in predicting mechanical response in many situations [5, 7]. These models need to take into consideration many complicated processes that paperboard exhibits, some of which are related to the high degree of anisotropy, large deformations, creep, and non-linear elasticity. Due to the fact that difficult phenomena are difficult to model, constitutive models for predicting paperboard behaviour usually incorporate modelling parameters which have no physical meaning or which are difficult to measure. These need to be calibrated by comparing model predictions with reality. Furthermore, overlooked mechanisms in paperboard processes usually reveal themselves when doing quantitative and qualitative comparisons of empirical tests and predicted response. This motivates the need for measurements of paperboard and will continue doing so at least for as long as constitutive models need to be further refined.

2.4 Continuum mechanics

This section will briefly cover some fundamentals of continuum mechanics [8] as well as go through a few useful tensors which are often used to characterize changes in geometry of a specimen.

2.4.1 Undeformed and deformed configurations

A configuration of a specimen is a space which it occupies. It is convenient to define a reference configuration with material points denoted by $\mathbf{X}$, from which motion can be described. The reference configuration is usually the initial, undeformed configuration of the specimen. The current or deformed configuration is then the configuration at time t, with material points denoted as $\mathbf{x}$.

2.4.2 Lagrangian and Eulerian descriptions

The Eulerian description of motion is such that the current configuration is taken as the reference when describing motion of the specimen. These are essentially the motion described in a spatial coordinate system.

On the other hand, the Lagrangian description is such that the motion is described in terms of the reference configuration of the specimen.

2.4.3 Useful tensors

The displacement tensor is defined as

$$\mathbf{u} = \mathbf{x} - \mathbf{X}\,, \tag{2.1}$$

and it relates material points in current configuration with material points in the reference configuration with a translation.

From $\mathbf{u}$, the deformation gradient tensor, $\mathbf{F}$, which describes the rate of change of line segments between material points, can be obtained by

$$\mathbf{F} = \frac{\partial \mathbf{x}}{\partial \mathbf{X}} = \frac{\partial(\mathbf{u} + \mathbf{X})}{\partial \mathbf{X}} = \frac{\partial \mathbf{u}}{\partial \mathbf{X}} + \frac{\partial \mathbf{X}}{\partial \mathbf{X}} = \frac{\partial \mathbf{u}}{\partial \mathbf{X}} + \mathbf{I}\,. \tag{2.2}$$

The right Cauchy-Green deformation tensor, $\mathbf{C}$, is defined as

$$\mathbf{C} = \mathbf{F}^T\mathbf{F}\,, \tag{2.3}$$

which gives the square of changes in distances between material points. The Green strain tensor, defined as

$$\mathbf{E} = \frac{1}{2}\left(\mathbf{C} - \mathbf{I}\right)\,, \tag{2.4}$$

can now be calculated from the deformation gradient. $\mathbf{E}$ is a measure quantifying how much a local displacement differs from a rigid body displacement and is therefore of use for evaluating large deformations in specimens. For example, **E22** can be used to measure the out-of-plane (ZD) strain.

2.5 Shape functions in the finite element method

Mathematical models are often used together with the finite element method to allow for more complex simulations. The finite element method as such will not be covered here, but only the basic theory concerning shape functions of rectangular elements since this is required for performing texture mapping as done in the simulation rendering, section 3.3.6. See Chapter 7 of Ottosen & Petersson [9] for a more in-depth discussion.

Consider the element and a query point, q, in figure 2.5. We first note that we have

$$p_i' = (x_i', y_i'), \quad p_i = (x_i, y_i), \quad q' = (x', y'), \quad q = (x, y)\,. \tag{2.5}$$

Now, the shape functions for a rectangular element are defined as

$$N_1(x,y) = \frac{(x-x_2)(y-y_4)}{(x_1-x_2)(y_1-y_4)}\,, \tag{2.6}$$

$$N_2(x,y) = \frac{(x-x_1)(y-y_3)}{(x_2-x_1)(y_2-y_3)}\,, \tag{2.7}$$

$$N_3(x,y) = \frac{(x-x_4)(y-y_2)}{(x_3-x_4)(y_3-y_2)}\,, \tag{2.8}$$

$$N_4(x,y) = \frac{(x-x_3)(y-y_1)}{(x_4-x_3)(y_4-y_1)}\,. \tag{2.9}$$

$$\tag{2.10}$$

and they all satisfy

$$N_i(p_i) = 1, \quad N_i(p_j) = 0,\ j \neq i \tag{2.11}$$

In local coordinates we can now obtain

$$x' = \mathbf{N}\mathbf{a}_x = [N_1(q)\ N_2(q)\ N_3(q)\ N_4(q)] \begin{bmatrix} x_1' - x_1 \\ x_2' - x_2 \\ x_3' - x_3 \\ x_4' - x_4 \end{bmatrix}, \tag{2.12}$$

and

$$y' = \mathbf{N}\mathbf{a}_y = [N_1(q)\ N_2(q)\ N_3(q)\ N_4(q)] \begin{bmatrix} y_1' - y_1 \\ y_2' - y_2 \\ y_3' - y_3 \\ y_4' - y_4 \end{bmatrix}, \tag{2.13}$$

thus mapping the point q in the undeformed element to the point q' in the deformed element in local coordinates.

Consider now that we have a bitmap texture with a grayscale intensity defined at a number of query points in an undeformed element; we can now translate all these points to their corresponding position in the deformed element, thus obtaining intensity values at known points in the deformed element. Inbetween these points one can then perform interpolation such that a texture covering the undeformed element is mapped to the deformed element.

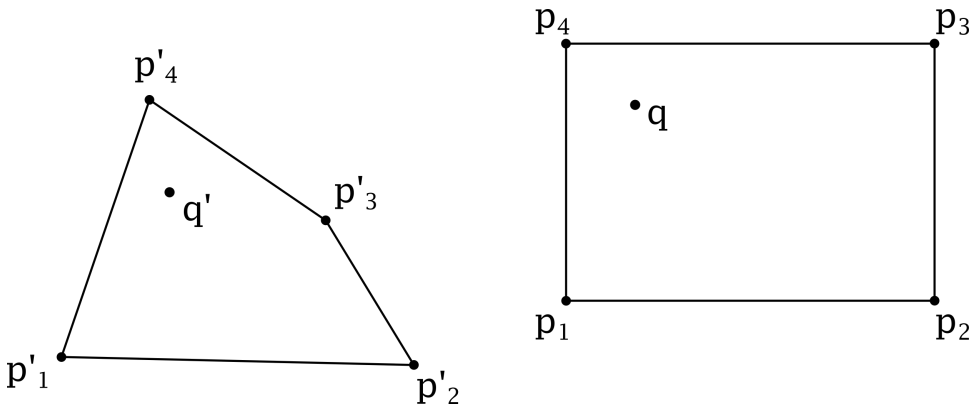

Figure 2.5: An element in its undeformed and deformed shape with a query point in both configurations. *Right*: Undeformed element in its reference configuration. *Left*: Deformed element in the current configuration.

2.6 Optical flow

Optical flow refers to the perception of flow of motion in a sequence of images. Solving the optical flow problem means finding the flow, $\vec{V}$, which translates a material point from one frame at time t, to another frame at time $t + \Delta t$.

To find the optical flow, one must first determine what constitutes the relation between *real motion* and *apparent motion*. The following two assumptions solidify this relation: first, if an object has undergone a small spatial translation during the time Δt, then it can be assumed that a point in the image plane belonging to the object has been translated by $(\Delta x, \Delta y)$. Secondly, if temporal resolution is high, it can be assumed that the intensity of a point belonging to the object has the same intensity in both frames.

Note the somewhat conflicting nature of these two points. Consider for example a blank sphere with uniform texture rotating around a stationary point in space. The rotation will not influence how the sphere is perceived since the shading will not change, and since the sphere is blank, there will be no occlusion. The apparent motion of all points on the sphere will be zero, while the real motion of these points will be non-zero.

Now, to derive the optical flow constraints [10], it is first stated that

$$I(x, y, t) = I(x + \Delta x, y + \Delta y, t + \Delta t)\,, \tag{2.14}$$

where $I(x, y, t)$ is the intensity at a point in the image with coordinates (x, y), at time t. This equation is commonly referred to as *the brightness constancy equation*.

If it is assumed that the movement is small between each frame then from equation 2.14, one can, by doing a Taylor series expansion, get

$$I(x + \Delta x, y + \Delta y, t + \Delta t) \approx I(x, y, t) + \frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\Delta t\,. \tag{2.15}$$

Combining 2.14 and 2.15 yields

$$\frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\Delta t \approx 0\,. \tag{2.16}$$

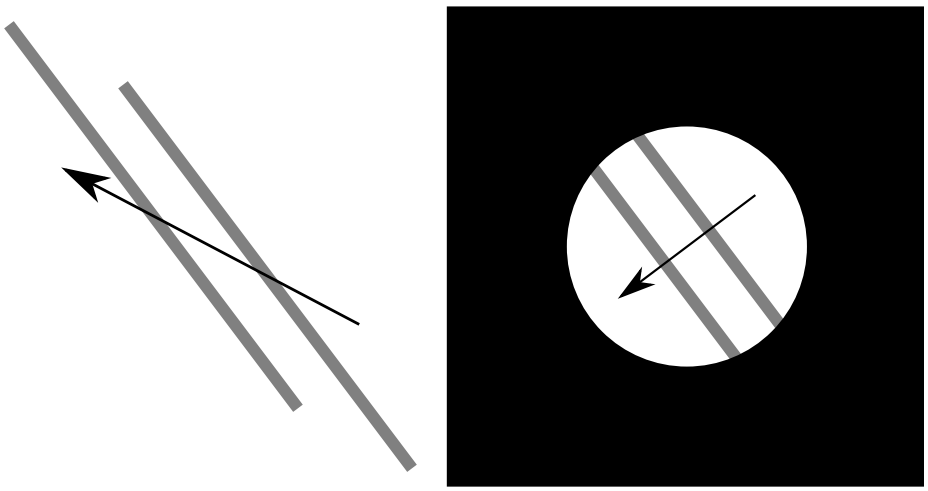

Figure 2.6: The two bars to the left are translated according to the left vector. By covering the end tips of the bars, the perceived motion of the bars is that they are moving according to the right vector, even though the real motion is according to the left vector.

Dividing by Δt then gives

$$\frac{\partial I}{\partial x}\frac{\Delta x}{\Delta t} + \frac{\partial I}{\partial y}\frac{\Delta y}{\Delta t} + \frac{\partial I}{\partial t}\frac{\Delta t}{\Delta t} = \frac{\partial I}{\partial x}V_x + \frac{\partial I}{\partial y}V_y + \frac{\partial I}{\partial t}\,.$$

We thus have

$$\begin{bmatrix} \frac{\partial I}{\partial x} & \frac{\partial I}{\partial y} \end{bmatrix} \cdot \begin{bmatrix} V_x & V_y \end{bmatrix}^T = -\frac{\partial I}{\partial t}\,,$$

or

$$\nabla I^T \cdot \vec{V} = -I_t\,, \tag{2.17}$$

where $\nabla I^T = [\partial I/\partial x,\ \partial I/\partial y]$, is the spatial intensity gradient of the image. This equation in two unknowns for every point is known as the *optical flow constraint equation*. It cannot be solved as is, which is due to the *aperture problem*, illustrated in figure 2.6. Consider now the case where we study a series of images in a video, and where the frame-rate is constant. We then have that Δt is constant. We can thus treat t as the index denoting image number t in a sequence of images. In this specific setting, the brightness constancy constraint is thus

$$I(x, y, t) = I(x + \Delta x, y + \Delta y, t + 1)\,. \tag{2.18}$$

By doing a Taylor series expansion like before, we get

$$I(x + \Delta x, y + \Delta y, t + 1) \approx I(x, y, t) + \frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\,, \tag{2.19}$$

and by combining equations 2.18 and 2.19 in the same manner as in the general case above, we again get that

$$\nabla I^T \cdot \vec{V} = -I_t \,. \tag{2.20}$$

In the proposed method we thus constrain ourselves to the above situation.

2.6.1 Optical flow approximation methods

Due to the aperture problem one usually needs to add further constraints and assumptions. Lucas and Kanade [11] assume that neighboring pixels in an image have a similar motion and that the motion is small between frames. By introducing the neighborhood constraint, a weighted least squares fit of the optical flow constraint equation can be stated as

$$\sum_{x,y \,\in\, \Omega} W \left[\frac{\partial I}{\partial x} V_x + \frac{\partial I}{\partial y} V_y + \frac{\partial I}{\partial t} \right]^2 , \tag{2.21}$$

where Ω is a neighborhood of pixels and where W is a matrix containing weights which favour pixels near the center of the neighborhood. Minimizing this expression yields an approximation of $\begin{bmatrix} V_x & V_y \end{bmatrix}$. The solution to the weighted least squares minimization problem is

$$\begin{bmatrix} \sum W \frac{\partial I^2}{\partial x} & \sum W \frac{\partial I}{\partial x}\frac{\partial I}{\partial y} \\ \sum W \frac{\partial I}{\partial y}\frac{\partial I}{\partial x} & \sum W \frac{\partial I^2}{\partial y} \end{bmatrix} \begin{bmatrix} V_x \\ V_y \end{bmatrix} = - \begin{bmatrix} \sum W \frac{\partial I}{\partial x}\frac{\partial I}{\partial t} \\ \sum W \frac{\partial I}{\partial x}\frac{\partial I}{\partial t} \end{bmatrix} . \tag{2.22}$$

This is a sparse optical flow estimation method which is typically used to track the movement of selected points in an image, in contrast to dense optical flow estimation in which every pixel in an image has an associated estimated flow vector. By employing a pyramidal representation of an image in which the raw image is down-sampled for each level in the pyramid, and applying the method at each level in the pyramid, larger displacements can be captured [12].

A popular dense optical flow estimation method is Farnebäcks two frame motion estimation [13]. The key concept of the method is based on polynomial expansion which is the idea of approximating a neighborhood of pixels by a polynomial

$$f(\mathbf{x}) \sim \mathbf{x}^T \mathbf{A} \mathbf{x} + \mathbf{b}^T \mathbf{x} + c \,, \tag{2.23}$$

where the coefficients, $\mathbf{A}$, $\mathbf{b}$, and c are estimated using a weighted least squares scheme. The exact quadratic polynomial

$$f_1(\mathbf{x}) = \mathbf{x}^T \mathbf{A}_1 \mathbf{x} + \mathbf{b}_1^T \mathbf{x} + c_1 \,, \tag{2.24}$$

and its displacement

$$f_2(\mathbf{x}) = f_1(\mathbf{x} - \mathbf{d}) = \mathbf{x}^T \mathbf{A}_2 \mathbf{x} + \mathbf{b}_2^T \mathbf{x} + c_2 \,, \tag{2.25}$$

then yields a system of equations

$$\mathbf{A}_2 = \mathbf{A}_1 \,, \tag{2.26}$$

$$\mathbf{b}_2 = \mathbf{b}_1 - 2\mathbf{A}_1\mathbf{d} \,, \tag{2.27}$$

$$c_2 = \mathbf{d}^T\mathbf{A}_1\mathbf{d} - \mathbf{b}_1^T\mathbf{d} + c_1 \,, \tag{2.28}$$

in which the translation $\mathbf{d}$ can be solved for,

$$2\mathbf{A}_1\mathbf{d} = -(\mathbf{b}_2 - \mathbf{b}_1) \,, \tag{2.29}$$

$$\mathbf{d} = -\frac{1}{2}\mathbf{A}_1^{-1}(\mathbf{b}_2 - \mathbf{b}_1) \,. \tag{2.30}$$

The method allows for iterative, multi-scale dense optical flow estimation in which the displacement field is assumed to vary smoothly.

More modern methods use deep neural networks for estimating flow fields. FlowNet is one such network which uses convolutional neural networks for flow estimation [14]. The network takes two consecutive frames as an input and produces an optical flow field estimation as the output. These networks require vast amounts of ground truth optical flow fields to accurately estimate the flow, and since ground truth optical flow fields for real world situations are very difficult to obtain, synthetic datasets such as the MPI Sintel Flow Dataset [15] are used to train these networks.

LiteFlowNet is another such network which consists of an encoder which maps an image pair into a pair of feature pyramids, and a decoder which estimates the flow field, starting from the coarse features at the top of the pyramid and working its way down to finer details [16]. In contrast to state of the art networks such as the successor of FlowNet, FlowNet2 [17], which warp the first image in the pair towards the second, and then refine the optical flow estimation using feature maps extracted between the warped image and the first image, LiteFlowNet directly warps the feature maps of the second image. This allows for comparing feature-space distance instead of RGB distance.

Chapter 3
Method

First, in chapter 3.1 an overview of the method and the system will be given. In chapter 3.2 the theory behind the method is presented and in chapter 3.3 it is shown how the theory is put into practice and how the system is implemented.

3.1 Overview

The method uses optical flow estimation for tracking of material points between each frame. From this spatiotemporal data one can extract different measures. The deformation gradient, **F**, is of interest since it describes the rate of change of line segments between material points. Also the strain tensor, **E**, is of great importance. From **E**, we can get information about the strains in different directions in the material; for instance, from the out-of-plane strain one can quantify delamination, and given the spatiotemporal data obtained by following the material points using optical flow, delamination zones can be detected and measured temporally as well as spatially.

By reconstructing a current image by warping past images using the optical flow fields in between, one discovers that material intensity discontinuations cannot properly be reconstructed by warping. A structural difference between the reconstructed image and the real image is made apparent. This allows for localization, tracking, and quantification of openings in the material such as delamination.

To verify that the fields obtained with the method are correct, a folding simulation simulated in ABAQUS [18] has been processed in order to extract ground truth deformation fields. The simulation has then been rendered so that the part has texture, which allows for optical flow extraction for tracking the material points of the part. The fields obtained by the proposed method are then compared with the field obtained from the simulation.

Figure 3.1 shows a high level schematical overview of the system. Hereby follows a short overview of each part.

1. *Folding simulation*: A simulation of the folding process of an uncreased board in

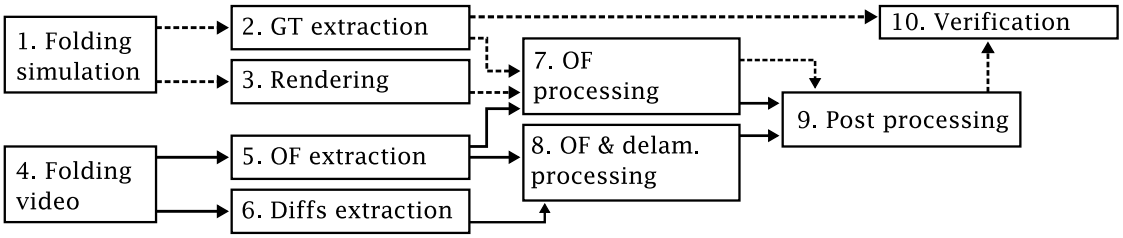

Figure 3.1: System overview. The dashed lines denote the pipeline for method verification while the full lines denote the pipeline for application of the method on video footage of the folding process.

ABAQUS. Data such as nodal positions for each step as well as the deformation gradient and different strain measures are obtained.

2. *GT extraction*: Ground truth measures for different fields are computed from the data of the folding simulation.

3. *Rendering*: Nodal positions of the simulated part for each time step are used for overlaying a fractal texture onto the part. Texture interpolation is computed with shape functions corresponding to a four-node quadrilateral element. A video is produced of the rendered simulation.

4. *Folding video*: Video footage of folding of a specified paperboard specimen.

5. *OF extraction*: Optical flow fields are extracted from a video. At the time of writing, LiteFlowNet is used for optical flow field computation.

6. *Diffs extraction*: Difference fields between a frame warped by applying the optical flow between it and the current frame, and the current frame, are computed.

7. *OF processing*: Optical flow fields are applied on a mesh following the motion of a specimen. Both the video at hand and the flow fields are needed for this since operations such as contour extraction of the specimen require raw video footage.

8. *OF & delamination processing*: Optical flow fields are applied on a mesh following the motion of a specimen which additionally breaks when nodes of the mesh are subject to a large difference at their position, as dictated by the difference fields. Here too, raw video data is required for further image analysis.

9. *Post processing*: Mesh nodal data is processed such that deformation fields and delamination quantities are acquired. Optionally, noise reduction and smoothing is performed.

10. *Verification*: Deformation fields as acquired from the simulation rendering can be compared with the ground truth deformation fields to validate the method.

3.2 Theory

Section 3.2.1 will explain how to reconstruct images using past optical flow fields. Section 3.2.2 will detail how the tracking of material points is done by application of the optical flow field, as well as solidify the connection between optical flow tracking and kinematics. Section 3.2.3 will go through how one can extract the deformation gradient and section 3.2.4 will explain the method of comparing warped and real images for finding delamination zones in a material.

3.2.1 Optical flow reconstruction

Let $I^{(t)}$ denote $I(x, y, t)$ and $I^{(t+1)}$ denote $I(x + \Delta x, y + \Delta y, t + 1)$. From 2.19 we get

$$I^{(t+1)} \approx I^{(t)} + \frac{\partial I^{(t)}}{\partial x}\Delta x^{(t)} + \frac{\partial I^{(t)}}{\partial y}\Delta y^{(t)} + \frac{\partial I^{(t)}}{\partial t} = I^{(t)} + \frac{\partial I^{(t)}}{\partial x}V_x^{(t)} + \frac{\partial I^{(t)}}{\partial y}V_y^{(t)} + \frac{\partial I^{(t)}}{\partial t}\,, \tag{3.1}$$

where $\partial I^{(t)}/\partial\{x, y, t\}$ denotes one of the partial derivatives of $I^{(t)}$, and where $V_x^{(t)}$ and $V_y^{(t)}$ denotes the optical flow components in the x and y directions respectively.

By including a remainder term for the Taylor approximation we get equality:

$$I^{(t+1)} = I^{(t)} + \frac{\partial I^{(t)}}{\partial x}V_x^{(t)} + \frac{\partial I^{(t)}}{\partial y}V_y^{(t)} + \frac{\partial I^{(t)}}{\partial t} + R^{(t)}\,. \tag{3.2}$$

We now collect the optical flow components and the gradients such that

$$I^{(t+1)} = I^{(t)} + L^{(t)} + R^{(t)}\,. \tag{3.3}$$

From this we see that for example

$$I^{(t+2)} = I^{(t+1)} + L^{(t+1)} + R^{(t+1)} = I^{(t)} + L^{(t)} + L^{(t+1)} + R^{(t)} + R^{(t+1)}\,. \tag{3.4}$$

So we can summarize the recursive relation as follows

$$I^{(t+k)} = I^{(t)} + \sum_{i=0}^{k-1} L^{(t+i)} + \sum_{i=0}^{k-1} R^{(t+i)}\,. \tag{3.5}$$

It is more convenient to be able to construct the current I from past ones and the optical flow of these, so, dropping the remainders for now, we instead say

$$I^{(t)} \approx I^{(t-k)} + \sum_{i=1}^{k} L^{(t-i)} = \tilde{I}^{(t,k)}\,. \tag{3.6}$$

3.2.2 Tracking of material points

Let $\mathbf{u}$ denote a displacement of any material point such that $\mathbf{u}^{(t-k,t)} = \mathbf{x}^{(t)} - \mathbf{x}^{(t-k)}$ for the material points in frame t and for the material points in frame $t - k$, respectively.

By selecting all material points such that $\mathbf{x}^{(t-k)}$ coincides with every pixel belonging to the material we get a regular mesh of pixels in an Eulerian description. By further selecting

the first frame such that $\mathbf{X} = \mathbf{x}^{(0)}$ is the undeformed reference configuration of the material, the material points will also form a regular mesh in a Lagrangian description. In reality, it will not be exactly regular since the paperboard might exhibit some deformations before folding such as the deformation caused by creasing.

By applying the optical flow to the regular pixel grid between frame 0 and 1 such that for every $I^{(0)}$ which belongs to the material we have

$$I(x, y, 0) = I(x + v_x, y + v_y, 1)\,, \tag{3.7}$$

and since the mesh is regular in both the Eulerian and Lagrangian description we get a displacement of the material point at position (x, y)

$$\mathbf{u}^{(0,1)} = \mathbf{x}^{(1)} - \mathbf{x}^{(0)} = \mathbf{x}^{(1)} - \mathbf{X}\,, \tag{3.8}$$

where $\mathbf{u}$ contains the x and y components of the optical flow vector at (x, y) such that

$$\mathbf{u}^{(0,1)} \approx V^{(0)}\,, \tag{3.9}$$

and where V contains the components of the optical flow field for every pixel belonging to the material.

We thus see that by applying the optical flow field on the pixel mesh of the undeformed reference configuration by displacing each pixel by adding V, we displace a material point by the same amount up to scale. The scale factor will be a constant with unit pixel/L where L is a length unit in for example which the thickness of the paperboard is expressed in.

Continuing like this we get

$$\begin{aligned} V^{(2)} &\approx \mathbf{u}^{(1,2)} = \mathbf{x}^{(2)} - \mathbf{x}^{(1)}\,, \\ V^{(3)} &\approx \mathbf{u}^{(2,3)} = \mathbf{x}^{(3)} - \mathbf{x}^{(2)}\,, \\ &\vdots \\ V^{(n)} &\approx \mathbf{u}^{(n-1,n)} = \mathbf{x}^{(n)} - \mathbf{x}^{(n-1)}\,. \end{aligned}$$

From the above we can express the tracking of the mesh for a whole sequence of frames from 0 to n with the recurrence relation

$$\begin{cases} \mathbf{x}^{(n)} = \mathbf{x}^{(n-1)} + \mathbf{u}^{(n-1,n)} \approx \mathbf{x}^{(n-1)} + V^{(n)}\,, \\ \mathbf{x}^{(0)} = \mathbf{X}\,. \end{cases} \tag{3.10}$$

We also acknowledge the following relationship

$$\mathbf{x}^{(t)} \approx \mathbf{x}^{(t-k)} + \sum_{i=1}^{k} V^{(t-i)}\,. \tag{3.11}$$

It is important to note that since pixels define a discrete grid of points and the flow field has real vectors at each pixel, subpixel interpolation needs to be performed. This is simply done by linearly interpolating the x and y components of the flow field V, in each dimension, with a material point's current real coordinates. See listing 3.4 for implementation details regarding the tracking of material points, including interpolation of the flow field.

3.2.3 Extraction of the deformation gradient

We know that the deformation gradient $\mathbf{F}$ is defined as

$$\mathbf{F} = \frac{\partial \mathbf{x}}{\partial \mathbf{X}}. \tag{3.12}$$

Letting $\mathbf{u}$ denote the displacement of any point such that $\mathbf{u} = \mathbf{x} - \mathbf{X}$ we instead have $\mathbf{x} = \mathbf{X} + \mathbf{u}$ which allows for the following construction

$$\mathbf{F} = \frac{\partial(\mathbf{X} + \mathbf{u})}{\partial \mathbf{X}} = \frac{\partial \mathbf{X}}{\partial \mathbf{X}} + \frac{\partial \mathbf{u}}{\partial \mathbf{X}} = \mathbf{I} + \frac{\partial \mathbf{u}}{\partial \mathbf{X}}. \tag{3.13}$$

This allows for expressing the deformation gradient either by the current configuration $\mathbf{x}$ which has to be obtained by adding the optical flow fields through all frames like seen in the previous section, or by explicitly differentiating the sum of the flow fields which belong to the material points with respect to the reference configuration. For the sake of completeness we go through both versions.

Since we have a discrete grid of material points we get

$$\mathbf{I} + \frac{\partial \mathbf{u}}{\partial \mathbf{X}} = \frac{\partial \mathbf{x}}{\partial \mathbf{X}} \approx \mathbf{I} + \frac{\Delta \mathbf{u}}{\Delta \mathbf{X}} = \frac{\Delta \mathbf{x}}{\Delta \mathbf{X}}. \tag{3.14}$$

For an explicit expression of $\mathbf{F}$, consider first the one dimensional case: we have the points $\mathbf{X}_1$ and $\mathbf{X}_2$ in the initial configuration and the points $\mathbf{x}_1$ and $\mathbf{x}_2$ in the current configuration. We have

$$\Delta \mathbf{x} = \mathbf{x}_1 - \mathbf{x}_2\,, \tag{3.15}$$

$$\Delta \mathbf{X} = \mathbf{X}_1 - \mathbf{X}_2\,, \tag{3.16}$$

and

$$\Delta \mathbf{u} = \Delta(\mathbf{x} - \mathbf{X}) = \Delta \mathbf{x} - \Delta \mathbf{X} = \mathbf{x}_1 - \mathbf{x}_2 - (\mathbf{X}_1 - \mathbf{X}_2)\,, \tag{3.17}$$

which gives

$$\frac{\Delta \mathbf{u}}{\Delta \mathbf{X}} = \frac{\mathbf{x}_1 - \mathbf{x}_2 - (\mathbf{X}_1 - \mathbf{X}_2)}{\mathbf{X}_1 - \mathbf{X}_2} \quad \text{and} \quad \frac{\Delta \mathbf{x}}{\Delta \mathbf{X}} = \frac{\mathbf{x}_1 - \mathbf{x}_2}{\mathbf{X}_1 - \mathbf{X}_2}. \tag{3.18}$$

Extending this to the two dimensional case, where each point has an x and y component, gives

$$\frac{\Delta \mathbf{u}_x}{\Delta \mathbf{X}_x} = \frac{\mathbf{x}_{1,x} - \mathbf{x}_{2,x} - (\mathbf{X}_{1,x} - \mathbf{X}_{2,x})}{\mathbf{X}_{1,x} - \mathbf{X}_{2,x}}, \tag{3.19}$$

$$\frac{\Delta \mathbf{u}_x}{\Delta \mathbf{X}_y} = \frac{\mathbf{x}_{1,x} - \mathbf{x}_{2,x} - (\mathbf{X}_{1,x} - \mathbf{X}_{2,x})}{\mathbf{X}_{1,y} - \mathbf{X}_{2,y}}, \tag{3.20}$$

$$\frac{\Delta \mathbf{u}_y}{\Delta \mathbf{X}_x} = \frac{\mathbf{x}_{1,y} - \mathbf{x}_{2,y} - (\mathbf{X}_{1,y} - \mathbf{X}_{2,y})}{\mathbf{X}_{1,x} - \mathbf{X}_{2,x}}, \tag{3.21}$$

$$\frac{\Delta \mathbf{u}_y}{\Delta \mathbf{X}_y} = \frac{\mathbf{x}_{1,y} - \mathbf{x}_{2,y} - (\mathbf{X}_{1,y} - \mathbf{X}_{2,y})}{\mathbf{X}_{1,y} - \mathbf{X}_{2,y}}, \tag{3.22}$$

and in the same manner

$$\frac{\Delta \mathbf{x}_x}{\Delta \mathbf{X}_x} = \frac{\mathbf{x}_{1,x} - \mathbf{x}_{2,x}}{\mathbf{X}_{1,x} - \mathbf{X}_{2,x}}, \tag{3.23}$$

$$\frac{\Delta \mathbf{x}_x}{\Delta \mathbf{X}_y} = \frac{\mathbf{x}_{1,x} - \mathbf{x}_{2,x}}{\mathbf{X}_{1,y} - \mathbf{X}_{2,y}}, \tag{3.24}$$

$$\frac{\Delta \mathbf{x}_y}{\Delta \mathbf{X}_x} = \frac{\mathbf{x}_{1,y} - \mathbf{x}_{2,y}}{\mathbf{X}_{1,x} - \mathbf{X}_{2,x}}, \tag{3.25}$$

$$\frac{\Delta \mathbf{x}_y}{\Delta \mathbf{X}_y} = \frac{\mathbf{x}_{1,y} - \mathbf{x}_{2,y}}{\mathbf{X}_{1,y} - \mathbf{X}_{2,y}}. \tag{3.26}$$

From the above we then have an explicit expression for the deformation gradient from frame 0 to frame n

$$\mathbf{F} = \begin{bmatrix} \dfrac{\Delta \mathbf{u}_x^{(0,n)}}{\Delta \mathbf{X}_x} + 1 & \dfrac{\Delta \mathbf{u}_x^{(0,n)}}{\Delta \mathbf{X}_y} \\ \dfrac{\Delta \mathbf{u}_y^{(0,n)}}{\Delta \mathbf{X}_x} & \dfrac{\Delta \mathbf{u}_y^{(0,n)}}{\Delta \mathbf{X}_y} + 1 \end{bmatrix} = \begin{bmatrix} \dfrac{\Delta \mathbf{x}_x^{(0)}}{\Delta \mathbf{X}_x} & \dfrac{\Delta \mathbf{x}_x^{(0)}}{\Delta \mathbf{X}_y} \\ \dfrac{\Delta \mathbf{x}_y^{(0)}}{\Delta \mathbf{X}_x} & \dfrac{\Delta \mathbf{x}_y^{(0)}}{\Delta \mathbf{X}_y} \end{bmatrix}. \tag{3.27}$$

From $\mathbf{F}$ we can get other fields as well, see theory section 2.4.3. Listing 3.5 shows how to perform this derivation in MATLAB.

3.2.4 Detection of structural differences

Consider now that we apply the optical flow to an image, $I^{(t-k)} = I(x, y, t-k)$ such that

$$I(x, y, t-k) \approx I(x + v_x, y + v_y, t)\,. \tag{3.28}$$

For $k > 0$ we thus get that

$$\begin{bmatrix} v_x & v_y \end{bmatrix} = \sum_{i=1}^{k} V^{(t-i)}\,. \tag{3.29}$$

Now let $X^{(t)}$ be all the x-coordinates of all image points of the image $I^{(t)}$, and let $Y^{(t)}$ be all the y-coordinates. We can now express the mapping from $I^{(t-k)}$ to $I^{(t)}$ as a linear translation of all points

$$\begin{bmatrix} X^{(t-k)} & Y^{(t-k)} \end{bmatrix} + \sum_{i=1}^{k} V^{(t-i)} \approx \begin{bmatrix} X^{(t)} & Y^{(t)} \end{bmatrix}, \tag{3.30}$$

which warps image $I^{(t-k)}$ such that we get the reconstruction $\tilde{I}^{(t,k)}$, see equation 3.6.

Consider now that we have a discontinuation of intensity between frame $t-k$ and t such as in figure 3.2. Since we have performed a warping of image $t-k$, we also need to interpolate the new intensity values in the reconstruction so as to reflect the translation of points, each with an intensity value attached to it. If two pixels next to each other with

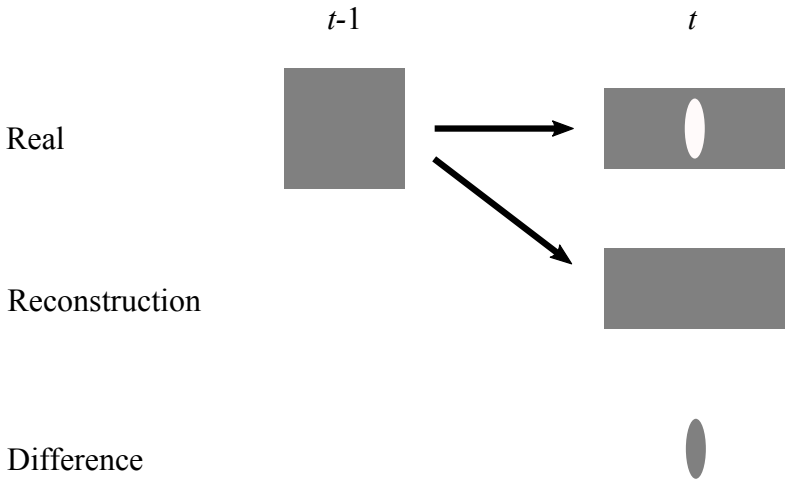

Figure 3.2: In the top row we have the actual image frames, $I^{(t-1)}$ and $I^{(t)}$. In the second row we have the reconstructed image $\tilde{I}^{(t,1)}$, made by warping $I^{(t-1)}$. The reconstruction cannot take into account the intensity change from $t-1$ to t which creates a structural difference between $I^{(t)}$ and $\tilde{I}^{(t,1)}$, shown in the bottom row.

the same intensity are separated a distance from each other as in stretching, the values in between those two pixels in the next frame will have to be interpolated, thus for for example linear interpolation, the intensity values inbetween will be the same as for the intensity values of the two pixels. There will thus be a structural difference between $I^{(t)}$ and $\tilde{I}^{(t,1)}$, illustrated in figure 3.2.

3.3 Implementation

The essential parts of the system will be described and core code snippets will be presented. Details such as variable initialization, declarations, imports and so on will in most cases be omitted.

3.3.1 Optical flow extraction

A PyTorch implementation of LiteFlowNet [19] is used for extraction of optical flow fields from the videos. A separate program written in Python goes through a video and feeds two consecutive frame tensors into the neural net, which then returns the estimated flow field. The tensors are saved for later use in other modules. Generators are heavily used in the Python modules as a simple tool for sequentially returning image frames from a video. See listing 3.1 for a simple generator using OpenCV for video parsing.

```
def imageGenerator(vidPath):
    cap = opencv.VideoCapture(vidPath)
    while True:
        ret, frame = cap.read()
        if frame is None:
            return
```

```
        yield frame
```

Listing 3.1: Simple image generator returning frames from a video in chronological order.

A code snippet of the generator in use for extracting flow fields can be see in listing 3.2. The variable `flow` will contain the x and y components of the flow field $V^{(i)}$ where $V^{(i)}$ is the flow between frames `prv` and `nxt`. Saving of flow fields, batch processing of flow fields, frame counting, and other implementation details are omitted.

```
prv = next(imGen)
for im in imGen:
    nxt = im
    flow = calcFlow(prv, nxt)
    prv = nxt
```

Listing 3.2: Intra-frame flow extraction for a video where `imGen` is an image generator, and `calcFlow` is a function calling any optical flow estimator.

3.3.2 Detection and extraction of structural differences

The structural similarity index [20] is used for comparing a reconstructed frame with the current frame. See listing 3.3. `nBack` is how many images back one wants to make the reconstruction from. It corresponds to k in the equations, see for example equation 3.29. In the `for` loop at lines 10-16 the warping is done, making the reconstructed image $\tilde{I}^{(t,k)}$. This essentially corresponds to equation 3.30. At line 16 we perform the remapping and it is here where we specify the interpolation scheme used when warping. The differences are saved in `allDiffs` and subsequently written to the hard-drive.

```
1  prv = next(imGen)
2  for nxt in imGen:
3      flow = np.array(flows[...,ii],dtype=np.float32)
4
5      if len(nBackFrames) == nBack:
6          del nBackFrames[0]
7      nBackFrames.append((prv,flow))
8
9      res = nBackFrames[0][0]
10     for i in range(len(nBackFrames)):
11         f = nBackFrames[i][1]
12         h, w = f.shape[:2]
13         f = -f
14         f[...,0] += np.arange(w)
15         f[...,1] += np.arange(h)[:,np.newaxis]
16         res = cv.remap(res, f, None, cv.INTER_LINEAR)
17     score, diff = compare_ssim(res, nxt, full=True)
```

```
18      allDiffs.append(diff)
19
20      prv = nxt
21      ii += 1
```

Listing 3.3: Extraction of structural differences between frames.

3.3.3 Application of optical flow and difference fields

MATLAB is used for applying the optical fields on a mesh which aims to follow the geometry of the specimen when folded. As described in section 3.2.2 the mesh is comprised of the x and y coordinates of the material points which initially are assumed to be on a regular grid. These points are then translated according to the optical flow field while also sub pixel interpolation is performed over the displacements. The structural difference fields are used to remove nodes from the mesh which are superimposed on a difference larger than a threshold, `diffs_t`. Listing 3.4 shows how the above is implemented.

`X0` and `Y0` contains the initial x and y coordinates, respectively, of the specimen. `interp2` is used to perform linear interpolation of the grid `xg`,`yg` for which every node has a defined optical flow vector with components `vy`, `vx`. `cntr` is a binary mask which defines where the material is in the image and it is obtained by thresholding the video frame in different ways. `X_clean` and `Y_clean` are used for post processing of the deformation fields. They are `NaN` where the nodes of the mesh lie outside of the contour mask. `X_brk` and `Y_brk` are `NaN` where the structural difference has been large at any point in the past and current frames. This can be thought of as integrating the differences such that a delamination area is formed, allowing for inspection of delamination in any frame.

```
[X0,Y0] = meshgrid(xlims(1):xlims(2),ylims(1):ylims(2));
while hasFrame(videoReader)
    diffs = diffmap(:,:,ind);
    diffs = diffs > diffs_t;
    vx    = flows(:,:,1,ind);
    vy    = flows(:,:,2,ind);

    X1 = interp2(xg,yg,vy,X0,Y0) + X0;
    Y1 = interp2(xg,yg,vx,X0,Y0) + Y0;

    [X_clean,Y_clean] = on_contour(X1,Y1,cntr);
    [X_brk,Y_brk]     = on_contour(X1,Y1,diffs);

    ind = ind + 1;

    X0 = X1;
    Y0 = Y1;
end
```

Listing 3.4: Application of optical flow fields to the frames of a video, additionally marking nodes where large structural differences have occurred.

3.3.4 Deformation gradient computation

In MATLAB we can compute the components of **F** simply as in listing 3.5. We get the change in x and y direction with respect to the x coordinates of the regular reference configuration mesh and likewise we get the change in x and y direction with respect to the y coordinates of the same mesh. This, and the fact that the mesh is in matrix index ordering rather than Cartesian index ordering explains the output ordering of the components of **F**.

```
[F22,F21] = gradient(X);
[F12,F11] = gradient(Y);
```

Listing 3.5: Computing the components of the deformation gradient, **F**, from the x coordinates, X, and y coordinates, Y, of the mesh.

Functions for retrieving other fields such as strain are omitted since given the equations in section 2.4.3 it is only a matter of performing matrix multiplication for each node in the mesh with a defined deformation gradient.

3.3.5 Ground truth extraction and computation

A simple folding simulation has been performed in ABAQUS. One then reports data such as nodal positions, the deformation gradient in each Gauss point, strain, and part topology, which is then processed in MATLAB with additional help of CALFEM toolbox, which is a finite element toolbox for MATLAB [21].

3.3.6 Rendering of simulation

The CALFEM toolbox is used as a framework for keeping track of the node positions over the steps of the simulation. A fractal Perlin noise texture image is created [22] and cropped up into smaller pieces matching the element size of the simulated part. Each such small piece is then interpolated using the shape functions of a 4-node rectangular element for finding a point to point correspondence between the undeformed element and the deformed element for any point inside the element, see listing 3.6 for shape functions and point mapping definitions. Note that in contrast to the shape function mapping of a point in a local coordinate system as is shown in section 2.5, the coordinates of the query point need to be added to the result of the local mapping in order for a global point mapping to be obtained. It is acknowledged that the use of function handles is not ideal when it comes to performance - simplicity and readability is the main motivation for doing it this way.

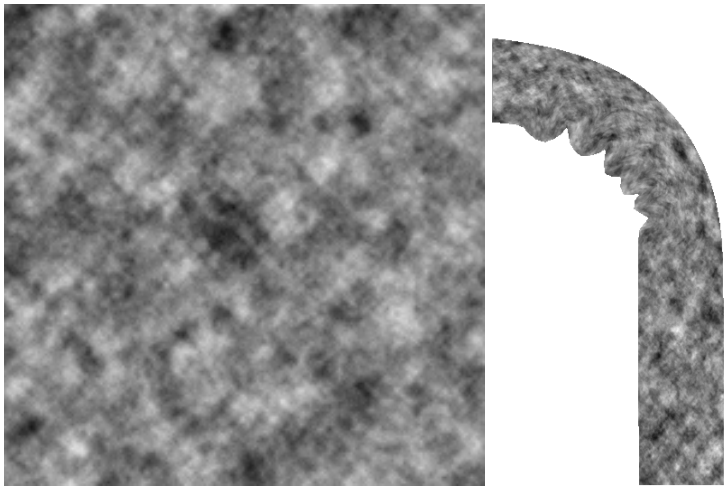

Figure 3.3: *Left*: Fractal Perlin noise texture. *Right*: Single frame of rendering of a folding simulation.

Figure (3.3) shows the Perlin noise fractal texture as well as a frame of the finished rendering. The choice of using a fractal Perlin noise as a texture is due to it being simple to implement while it exhibits randomness at different scales, making for good tracking features.

```
N1 = @(x,y) ((x-x2)*(y-y4))/((x1-x2)*(y1-y4));
N2 = @(x,y) ((x-x1)*(y-y3))/((x2-x1)*(y2-y3));
N3 = @(x,y) ((x-x4)*(y-y2))/((x3-x4)*(y3-y2));
N4 = @(x,y) ((x-x3)*(y-y1))/((x4-x3)*(y4-y1));

N = @(x,y) [N1(x,y);N2(x,y);N3(x,y);N4(x,y)];

map_q = @(N,q0,p0,p1) q0 + [N(q0(1),q0(2))'*(p1(:,1)-p0(:,1)),...
                            N(q0(1),q0(2))'*(p1(:,2)-p0(:,2))];
```

Listing 3.6: Shape function and mapping function definitions for a query point mapping, `q0`, from an undeformed rectangular element, `p0`, to a deformed element `p1`.

Chapter 4

Results

Ground truth comparisons will be presented for verification, as well as deformation fields, delamination and delamination zone area by angle, for a few folding videos. An analysis of total delamination area by crease depth and by paperboard will also be included. Furthermore, visualization of mesh following material points will be included to allow for qualitative evaluation of tracking capabilities. The optical flow is estimated with LiteFlowNet trained on the Sintel dataset.

There are 127 video samples in total, with 6 different paperboards and 3 different crease depths. A full table detailing the dataset can be seen in Appendix A. Depicted in the videos are paperboard samples being folded from their creased or uncreased reference configuration to a 165 degree folding angle and then relaxed to a 90 degree folding angle. The videos have a resolution of 1280 times 1024 pixels and the average number of frames for each video is 120 frames. Figure 4.1 shows the start, middle, and end frames of a typical video.

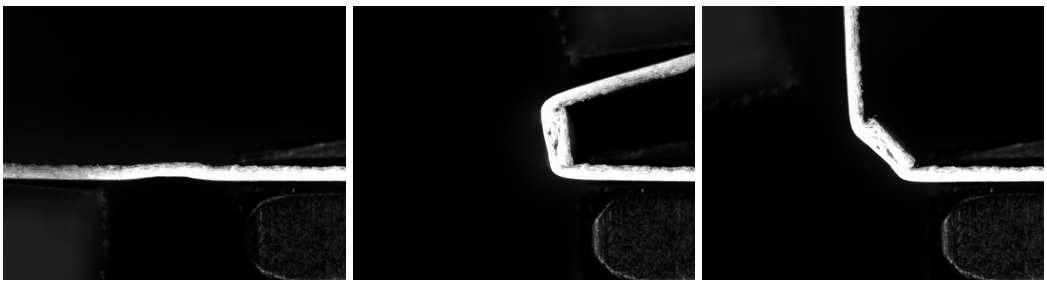

Figure 4.1: *From left to right*: Start frame of a video where the paperboard has not yet been folded, frame in the middle of a video where the paperboard has been folded to a 165 degree folding angle, end frame of a video where the paperboard has been relaxed back to a 90 degree folding angle.

4.1 Ground truth comparisons

The following is a comparison between the ground truth fields of the final configuration of the simulated material and the fields estimated by applying the proposed method on a rendering of the simulation. Ground truth field data in between nodes has been interpolated bilinearly. In addition, a smoothed image of the estimation using Fast Global Smoothing [23] has been included in the comparisons for all fields except the displacement fields.

Displacement fields **U1** and **U2** have been obtained by summing the nodal positions of the simulated part for each step. Figure 4.2 shows a full comparison.

The deformation gradient, **F**, has been obtained from ABAQUS for the Gauss points of the simulated part. The values in the nodes have been set to the deformation gradient value of a Gauss point belonging to the element of which the node is a vertex. Figure 4.3 shows a full comparison of the deformation gradient component **F11**.

From **F**, strain fields **E**, are obtained. A full comparison of the out-of-plane strain field **E22** can be seen in figure 4.4.

In figure 4.5 the mesh as defined when meshing the simulated specimen in ABAQUS can be compared with an illustration of the material point tracking of the render of the simulation by applying the proposed method.

4.2 Delamination

Results obtained in this section have been acquired by comparing $\tilde{I}^{(t,6)}$ with the current frame $I^{(t)}$ for calculating the differences with SSIM [20]. The difference fields have then been used for breaking up the pixel-by-pixel mesh following the material during the fold. The broken mesh for sample number 8 with a mesh node spacing of 2 pixels, and the full mesh with a pixel spacing of 4 pixels, as well as the initial mesh with 4 pixel spacing are all shown figure 4.9. Note that for illustration purposes, the mesh has a node spacing of 2 pixels and 4 pixels respectively, instead of 1 pixel spacing as is used normally.

Figure 4.6 shows the total delamination zone for 8 different samples at 165 degrees folding angle. Tabular 4.1 contains information about these and figure 4.7 shows plots of delamination area by folding angle for the samples.

Sample #	1	2	3	4	5	6	7	8
Board	A	B	B	D	C	A	B	E
Crease depth (mm)	0.1	0	0	0	0.1	0.2	0.1	0.2
Final area (10^3 px)	9.75	2.40	0.91	7.96	1.24	7.81	7.01	5.44

Table 4.1: Information about the 8 samples with board, crease depth, and final area as indicated by the red area in figure 4.6.

4.3 Delamination analysis

Total delamination zone area for all samples (see Appendix A for details) by crease depth can be seen in the box plot in figure 4.10. Additionally, mean delamination zone area over

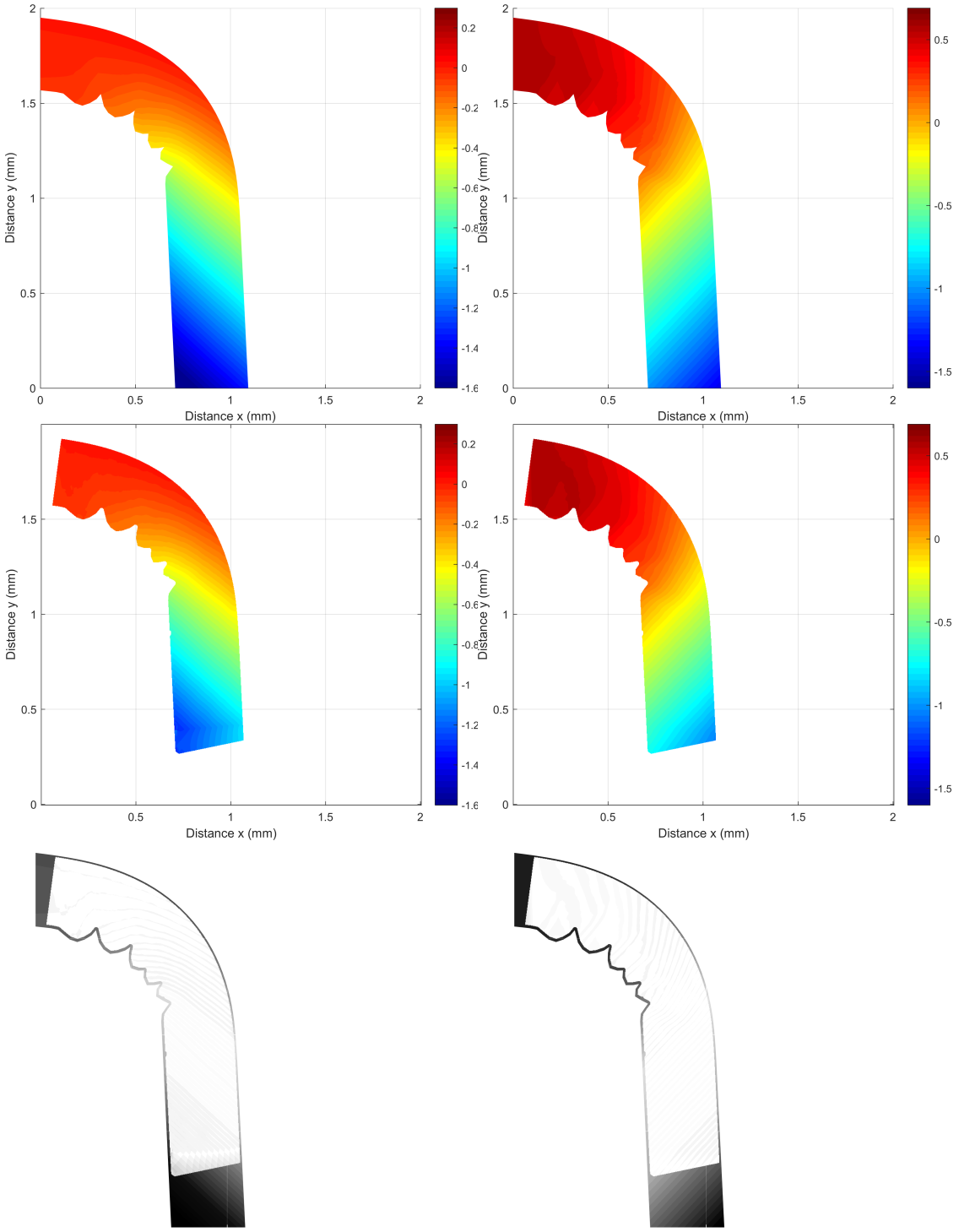

Figure 4.2: *Top row*: Ground truth displacement fields in x and y direction, **U1** and **U2**, respectively. *Left*: **U1**, *Right*: **U2**. *Middle row*: Estimated. *Bottom row*: Difference between ground truth and estimated.

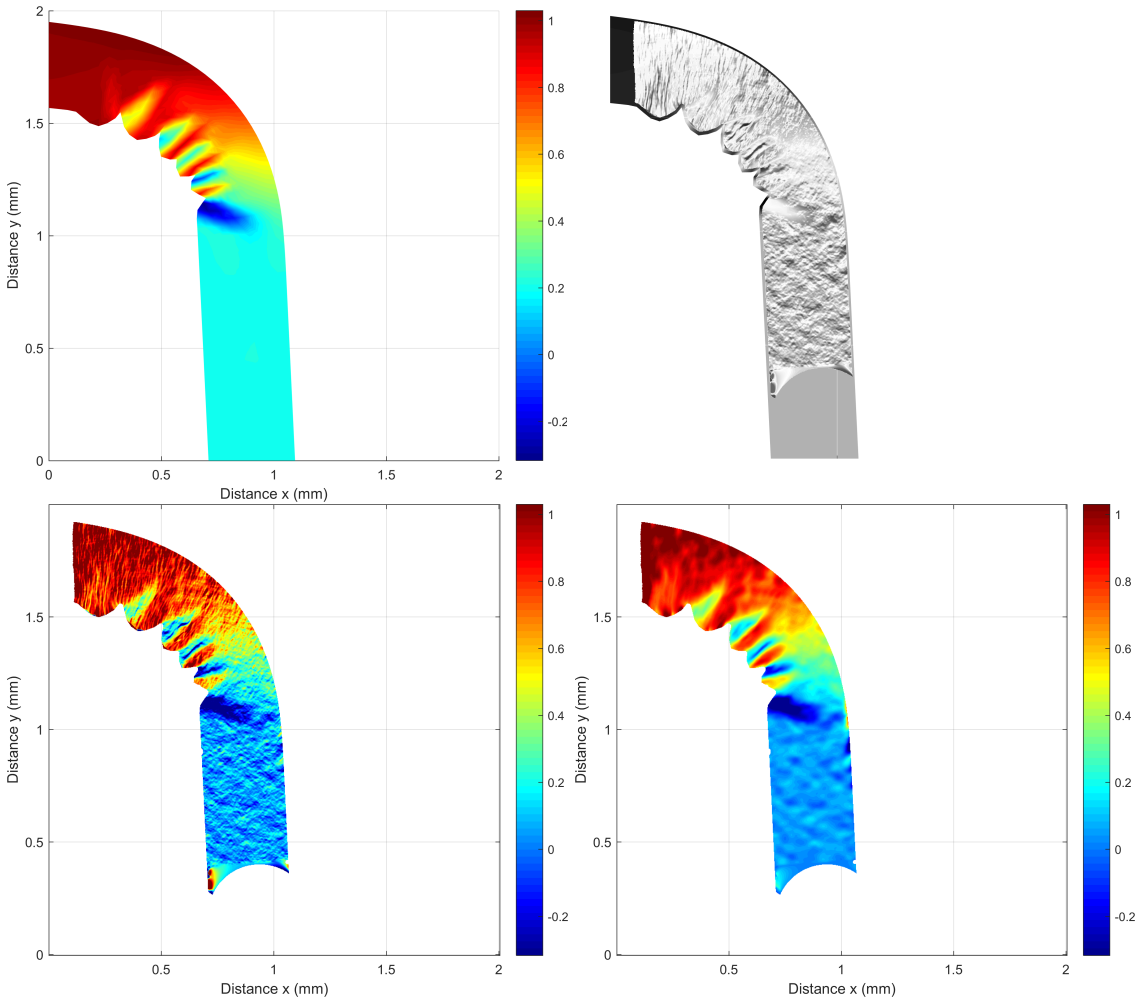

Figure 4.3: F11. *Top left*: Ground truth. *Top right*: Difference between ground truth and estimated. *Bottom left*: Estimated. *Bottom right*: Estimated with smoothing.

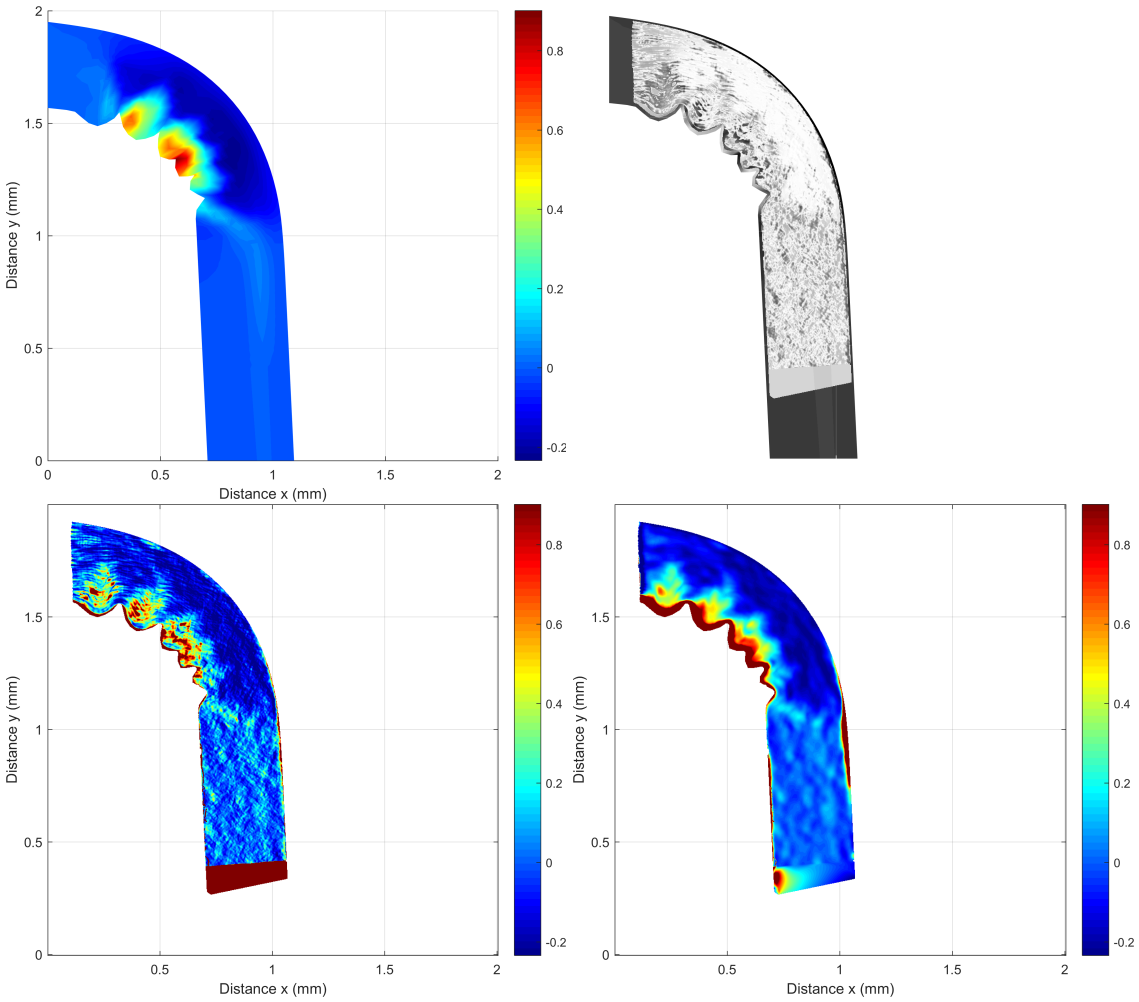

Figure 4.4: E22. *Top left*: Ground truth. *Top right*: Difference between ground truth and estimated. *Bottom left*: Estimated. *Bottom right*: Estimated with smoothing.

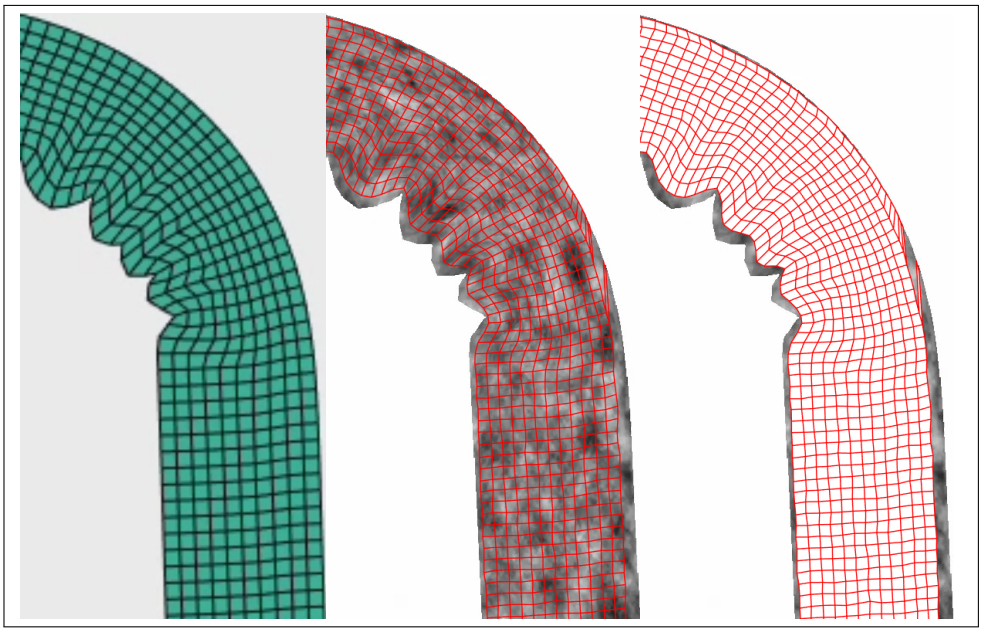

Figure 4.5: *Left*: Mesh as defined when meshing of the simulated specimen in ABAQUS. *Middle*: Mesh following the rendering of the specimen by application of optical flow fields. *Right*: Same as the middle image but with white face color for easier evaluation of correctness.

folding angle can be seen in figure 4.11 for all samples.

Figure 4.12 shows the same analysis as above but where the data is further divided into the different paperboards, to highlight the difference in area over angle response with respect to creasing.

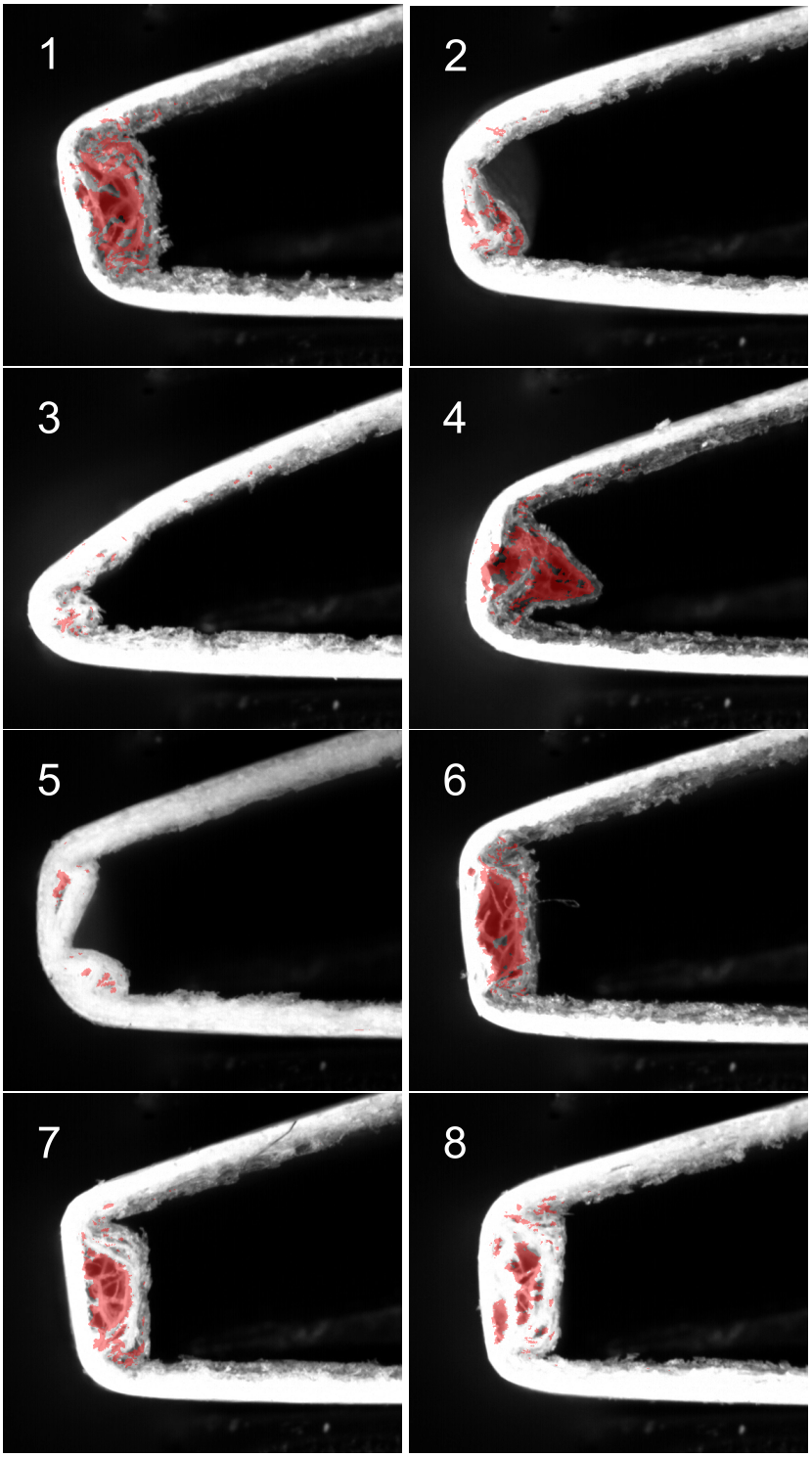

Figure 4.6: Delamination zone at 165 degrees folding angle obtained by tracking the mesh with removed nodes according to difference fields. The red area signifies the delamination zone.

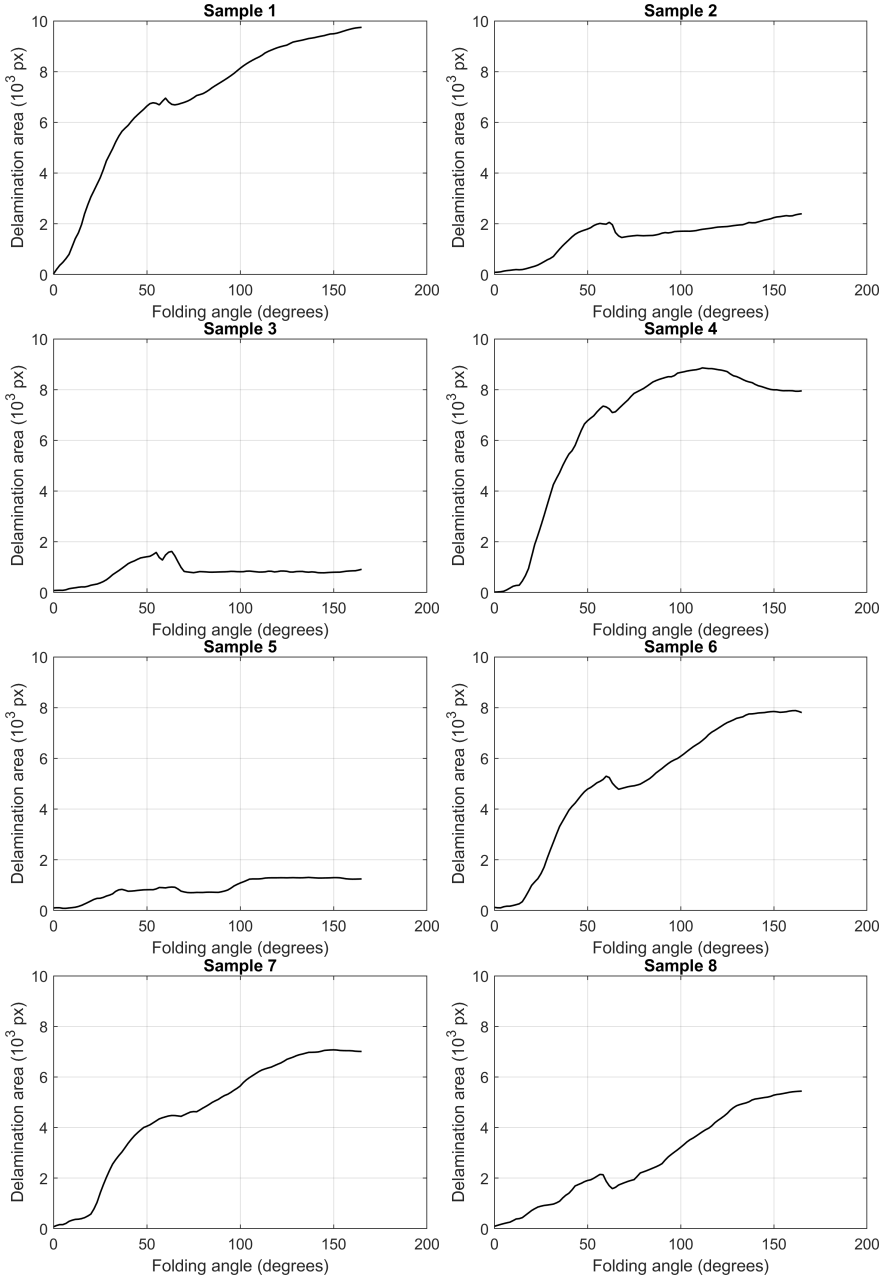

Figure 4.7: Delamination zone area by folding angle by counting the pixel area of the delamination zones obtained by tracking the mesh with removed nodes according to difference fields, for the eight samples.

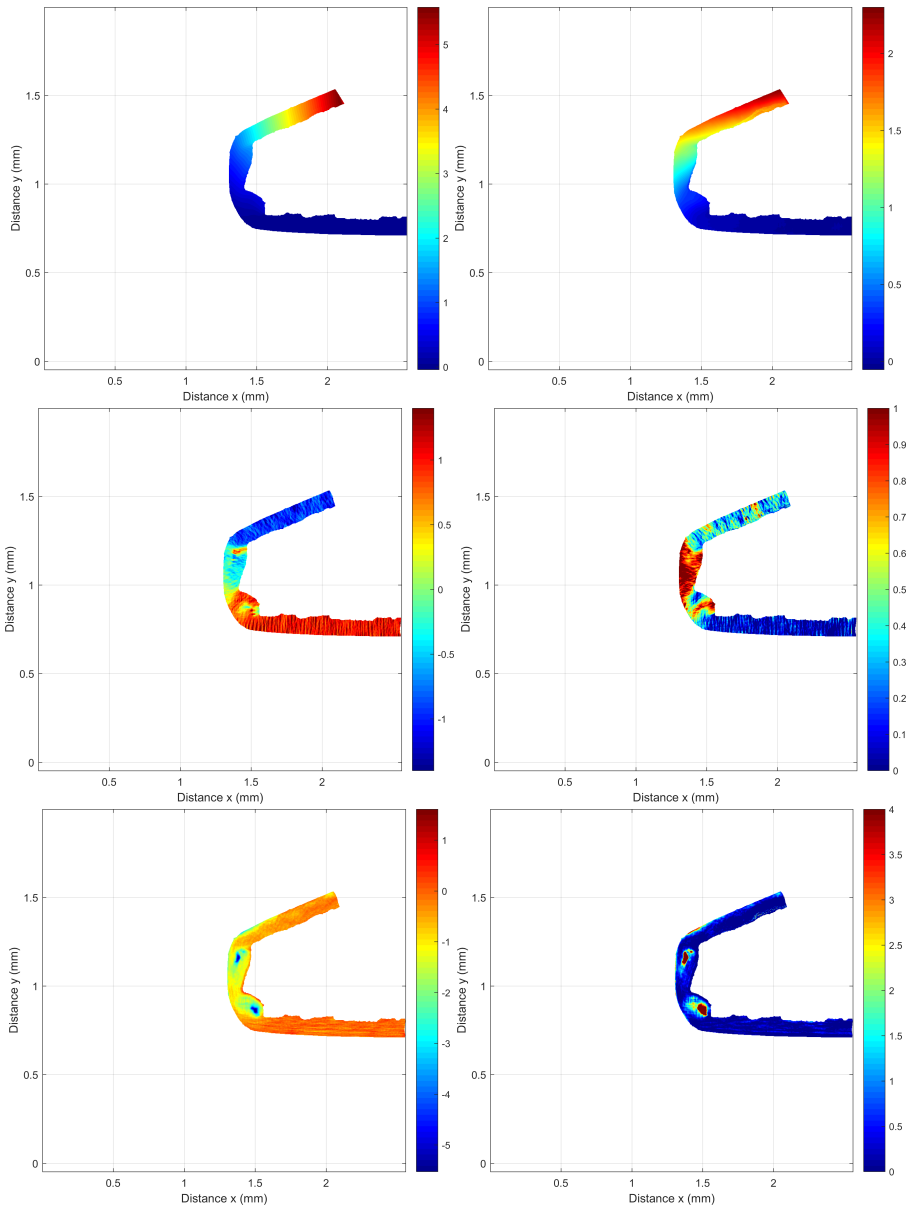

Figure 4.8: Fields for Sample number 5. *Top left*: **U1**. *Top right*: **U2**. *Middle left*: **F11**. *Middle right*: **F12**. *Bottom left*: **F21**. *Bottom right*: **E22**.

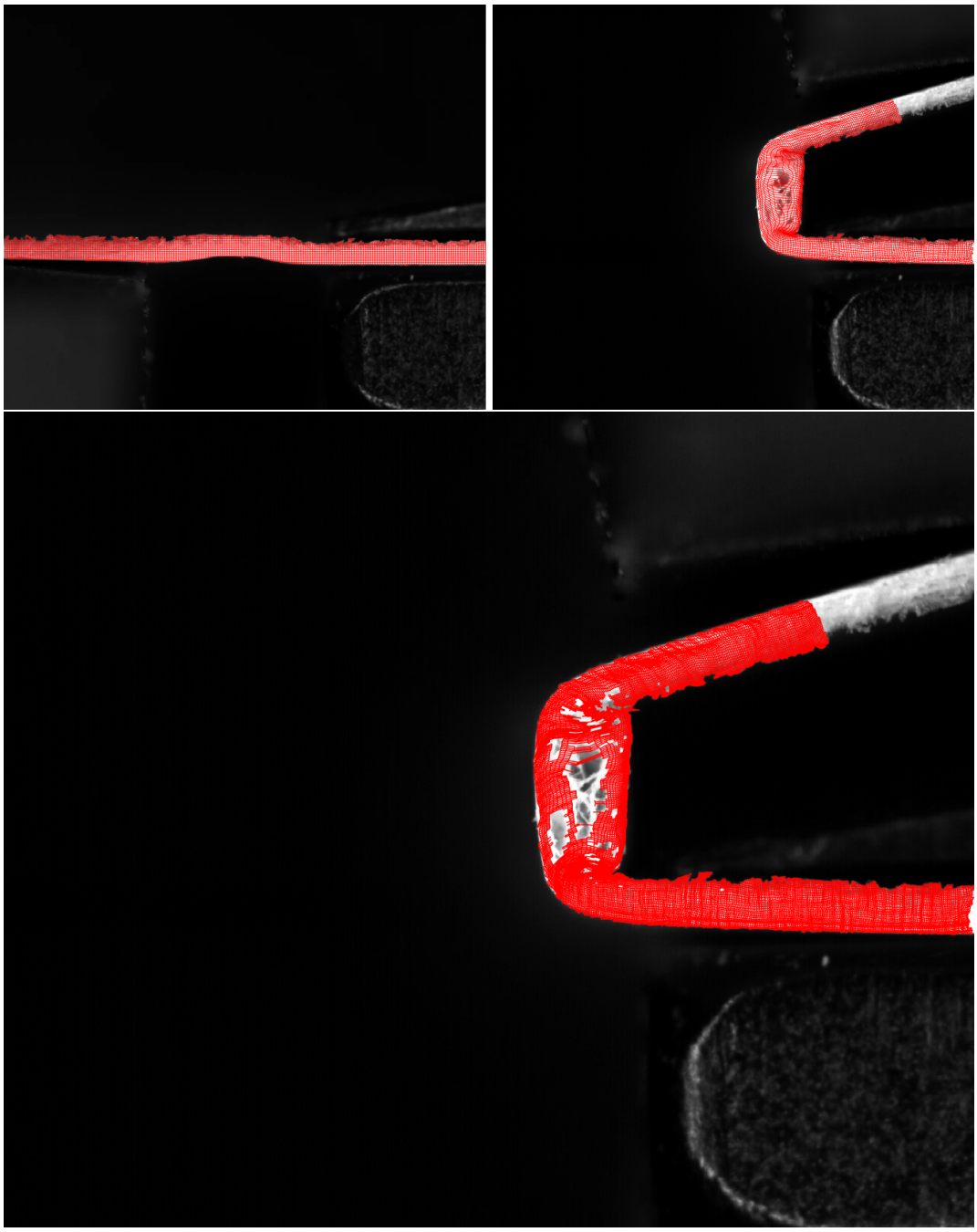

Figure 4.9: Mesh tracking material points. *Top left*: Initial mesh, 4 px spacing. *Top right*: Final mesh, 4 px spacing. *Bottom*: Final mesh with breaking, 2 px spacing.

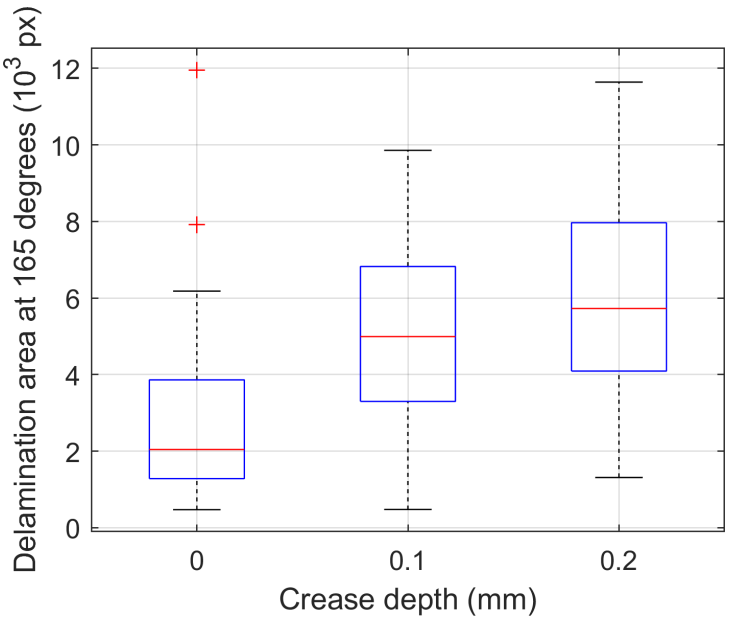

Figure 4.10: Final area in pixels of the delamination zone of each material by crease depth.

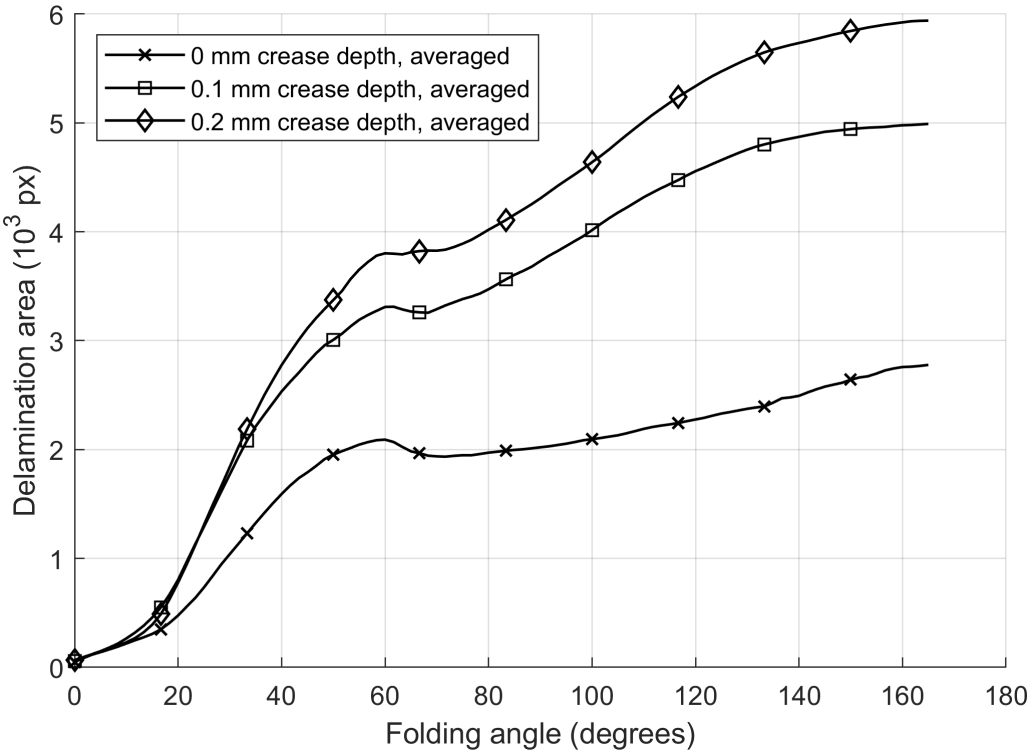

Figure 4.11: Mean area in pixels of the delamination zone by folding degree of samples by crease depth.

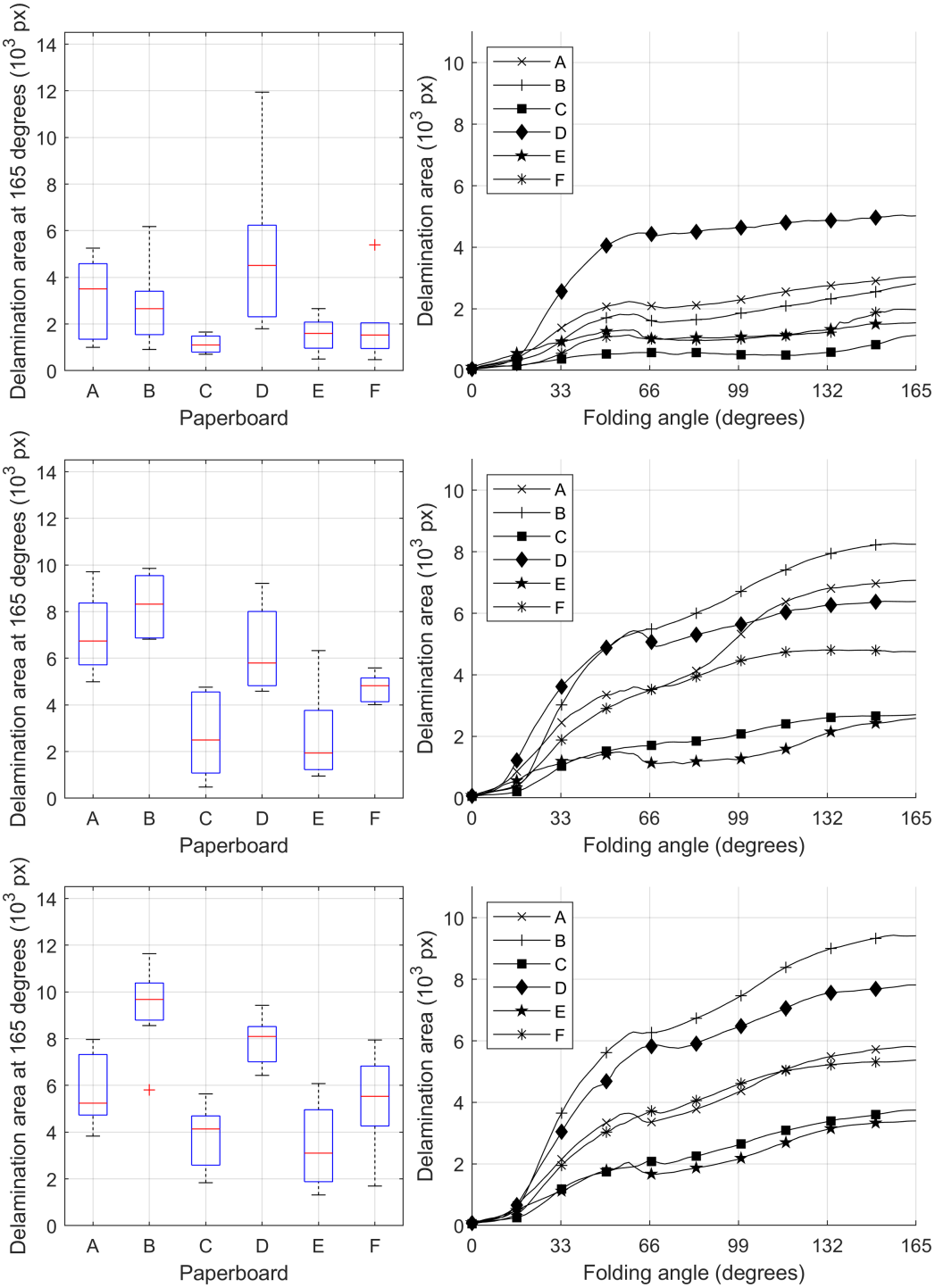

Figure 4.12: *Top row*: 0 mm crease depth. *Middle row*: 0.1 mm crease depth. *Bottom row*: 0.2 mm crease depth.

Chapter 5
Discussion

Here, a discussion of the method and the results will take place. Since much of the results are qualitative, the discussion for these will have a qualitative tendency. The quantitative results will be evaluated in a statistical manner, where suitable. Furthermore, paths not taken and future work will be discussed.

5.1 Method verification

It is fortunate that the nature of the problem at hand allows for comparison of ground truth data, easily obtained from simulation. However, whether the application of the method on a rendering of a simulation provides a good verification or not is difficult to assess, since a rendering still differs immensely from reality - especially the kind of rendering performed here, without taking into account physical phenomena such as motion blur.

As the tool for estimating the optical flow fields is LiteFlowNet trained on the Sintel dataset, the generalization performance of this neural network is the greatest with data matching that of the training data - and the rendering does not exhibit the same kind of motion or graphics which the Sintel database does.

A case could, analogous to the above, be made that the optical flow estimation would perform better on the actual videos of folding, and that the optical flow tracking of the rendering is sub-par of that of data more closely resembling the training data, which of course would be sensible.

5.1.1 Considerations

The Sintel dataset aims to provide realistic scenes suitable for training and comparing of optical flow estimation techniques, which is why features such as specular reflections, motion blur, defocus blur and atmospheric effects are rendered in the dataset. The rendering of the simulation, however, does not take into account any of these effects - it is

simply a sequence of rendered images put together in a video. As an example, motion blur isn't present in the rendering when usually this effect is strongly associated with motion in videos. This might make it a challenge to properly estimate the optical flow in the rendered video since it is quite different from the training data in the Sintel dataset, which is what the currently used network has been trained on.

It is also important to consider what the Perlin noise texture interpolation has done with the optical flow estimation. One might make the reflection that since the ground truth data is interpolated in the same way from the nodes of the specimen, this potential artifact should "even out". The difference is of course that with the rendering we are influencing the end position of the material points each time we interpolate the texture - if the interpolation leads to warping which exaggerates the motion of the specimen then these errors will accumulate over time.

5.1.2 Optical flow tracking

Qualitatively, the material point tracking by optical flow estimation can be evaluated by looking at the mesh in figure 4.5. Dense optical flow fields are interpolated with a certain spacing for illustration and qualitative evaluation purposes - creating the red mesh. Normally the mesh is finer: the flow field is queried in every pixel of the frame belonging to the material, yielding a 1 by 1 pixel mesh.

By looking at the mesh in figure 4.5 one can see that it seems to match the element mesh quite well. The wrinkles are fairly well captured as well as the general large motion of the specimen. Material points are lost at the edges and there is some noise in the lower region of the specimen - the grid is not quite regular.

5.1.3 Ground truth comparisons

As can be seen by looking at the field comparisons in section 4.1, the estimation matches quite well with the ground truth. As has been pointed out, the edges of the specimen are difficult to keep track of. This is likely in part due to the fact that optical flow estimation is generally difficult to perform at straight edges. Modern methods mitigate much of these problems but they can still be present, especially if the estimation network doesn't generalize from trained data to custom data in a desirable way, as has been discussed as a possible factor affecting ground truth comparisons.

One can also observe that the lower flat region of the specimen has been subject to some noise in the estimation. The noise is likely in the same order of magnitude everywhere on the rendering of the specimen but perhaps it is more visually pronounced in the flat region since the ground truth data shows a perfect smooth color. It can also be observed that in the light teal area of the ground truth field **F11** in figure 4.3 the value of the deformation gradient is at around 0.2 while in the estimated field it seems to be slightly lower as shown by the more blueish color in the estimations.

Overall, the features and tendencies of the ground truth fields are captured in the estimations, albeit not exactly. Smoothing the pixel by pixel estimations could provide for results that are more similar to the ground truth but this is akin to interpolating the flow field with a larger spacing than 1 pixel, leading to loss of detail in the process.

5.1.4 Summary of method verification

The proposed method captures trends and tendencies for the different fields which match quite well with the ground truth fields. However, it is difficult to know if this is a fair verification since the rendered simulation is very different from the data trained on by LiteFlowNet, and in fact also from the actual folding videos. Additionally the effect of bilinear texture interpolation on the optical flow has not been studied but it is acknowledged that this might affect the end-to-end material point error.

5.2 Delamination and deformation fields of actual samples

Due to the lack of ground truth data for the folding videos of the paperboards analyzed, the delamination analysis should be seen as an experiment. It will be more difficult to know what is good and what is bad, but observing the delamination zones overlaid on the paperboard as well as the material point mesh seems to give strong indications on whether the results are reasonable or not.

5.2.1 Optical flow tracking

Figure 4.9 shows the material point mesh (again, illustrated with a larger pixel spacing) when applying the proposed method on a folding video. The mesh breakage seems to match the actual delamination zones quite well and overall the mesh seems to have followed the material in a way which seems reasonable and which exhibits the expected warping.

Qualitative comparisons of the material point tracking capabilities of the optical flow estimation between the tracking of material points in the rendered simulation and the folding videos suggest that the tracking is more robust in the case of the folding videos, which is supported by the earlier discussion in section 5.1.

It should also be pointed out that a few different optical flow estimation techniques were used. Classical ones such as Horn & Schunk [24] and Lucas & Kanade [11] were tested, as well as more modern ones like Farnebäck [13].

5.2.2 Differences for estimating delamination

The difference maps stem from the fact that in some areas there is a spatio-temporal discontinuity of intensity, or rather the brightness constancy equation is undefined in these areas. Intensity changes between frames can in this way be captured momentarily before again being integrated into the current frame, while stepping forward through the frames of a video.

It had, for a while, been quite unclear how these momentary differences could be utilized in order to give meaningful measures. Placing points where the differences were high and then tracking these points in the same manner as tracking the material points was investigated. One could then perform some kind of adaptive flood filling from those points to find the delamination zones. The problem with this approach is that it is very

similar to just performing grayscale thresholding, which has a few undesirable characteristics. An example is that grayscale thresholding is very dependent on hyperparameters such as thresholding values for finding the outline of the specimen and for finding a "good" balance between the material brightness and the background brightness.

Ideally, the difference maps would help achieve the following:

- Spatio-temporal localization of delamination zones
- Spatio-temporal propagation of delamination zones
- Total delamination in final frame, seen as zones

It was by these points then decided that the removal of nodes in the material point mesh where the difference was at its maximum were to be performed. By employing this scheme, location in the specimen and per frame of each opening would be found by looking at where the nodes of the mesh are missing. Furthermore, since the mesh nodes are permanently removed, propagation of the delaminations zones are achieved by studying how the area where the nodes are missing changes. It then naturally follows from this that delamination zones are present in the final frame, allowing for analysis of zone characteristics such as the size and shape of it. This way of treating the mesh with the differences is essentially a way to integrate the differences by time, which is what was desired in the first place.

It is important to acknowledge the fact that the choice of k, determining which image $t - k$ to warp, impacts the results. A value of k which is too small might lead to the differences only capturing quick rips and cracks while disregarding slower openings in which the intensity gradient at the opening stretches over several frames. A value too large, however, will lead to the warped image being so different everywhere as to make the differences highlight false positives due to noise. A trade-off between detection accuracy and noise must be found. For the results, $k = 6$, which was found to be a good value, was used.

5.2.3 Delamination zones

As can be seen in section 4.2, zones seem to correspond well to where one would intuitively think that they would occur when seeing the image. Looking at figure 4.6 it is observed that zones for many different situations can be fairly accurately found. Small openings like seen in sample 5 and 8, as well as large ones are marked.

It is interesting to note what the result would be if one masked out everything in the image except the center bulge and then performed some kind of gray-scale thresholding - sample 6, 7, and 8 would look very different. Since fibers can be observed inside the delamination zone which have an intensity similar to the material surrounding the zone, these will, by thresholding, not be part of the opening. One would instead get many small areas instead of one large one.

Whether this is desirable or not is up to what is most practical. It might be best to consider fibers which are very strongly visually pronounced like in sample 6 and 7, but which maybe do not contribute as much to the ZD stiffness, as still being part of the delamination zone. This wouldn't be the case for grayscale thresholding.

5.2.4 Deformation fields

Material point mesh correspondence to actual deformation has been discussed and from making the point that the mesh seems to do a good job capturing the geometrical changes that the paperboard undergoes during folding it also follows that the fields obtained from the mesh should constitute reasonable values.

Sample 5 is shown in figure 4.8. By observing for example **E22** one can see that it seems to correspond well to where there actually is larger out-of-plane strains. Looking at the same sample in figure 4.6, this is also where the delamination zones are seen to be.

If these fields accurately capture the deformation of a sample then they allow for direct comparison of simulation predictions with what has actually happened, which would make it easier to calibrate models towards making accurate predictions of the folding process. It should of course be said that individual samples of the same paperboard can vary a lot in how they behave, which is why many such fields would need to be obtained for each paperboard. This is why automatic extraction of these without the need for much parameter tuning has been an important feature of the proposed method.

5.2.5 Statistical analysis

Figure 4.10 shows what is expected: a deeper crease leads to a larger delamination zone while an uncreased sample will have a much smaller zone and often almost no zone at all. One of the outliers at crease depth 0 mm can be seen in figure 4.6, sample 4. There is no flat bulge as seen in samples 1, 6, 7, and 8, resulting from a crease, but a sharp point instead. Obviously, the sample has large delamination but looking at the shape of the zone one can discriminate between this one and "proper" delaminations as the result of creasing. This suggests analyzing not only the total area but also the shape of the zone.

Looking at figure 4.7 and figure 4.11 one notes behaviour that has been observed by looking at delamination zones manually: the area increases rapidly up to some number and then at folding angle ϕ_0 the derivative of the area with respect to the angle is 0, after which the area often slowly rises again. This local maxima is even more pronounced when looking at the averages of all measurements in figure 4.11. What is interesting is that ϕ_0 doesn't seem to depend on crease depth and has a mean of about 56 degrees when evaluating over all samples. Even in the individual samples in figure 4.7, the same mean of 56 degrees seems quite close to what is observed.

A theory as to why the area starts rising again after ϕ_0 is that the zone at this point is often elongated and once the folding angle increases even more, the zone becomes rounder and thus the area could increase without the sample actually having a larger degree of delamination. This is also what has been personally observed in the folding videos.

In figure 4.12 is the same analysis as done for the crease depths but further divided into different paperboards. Here we note several things. Firstly, it seems that ϕ_0 is unequally pronounced in the paperboards, especially for crease depths of 0.1 mm and 0.2 mm. We see that for example ϕ_0 is difficult to observe in paperboards C and F - there's a small bump, but much less noticeable than for other paperboards.

Also note that paperboard D has a much larger delamination zone area at 0 mm crease depth than the other paperboards. Many of the samples of paperboard D exhibit the kind of behaviour seen in figure 4.6, sample 4, discussed earlier. Another observation to bring

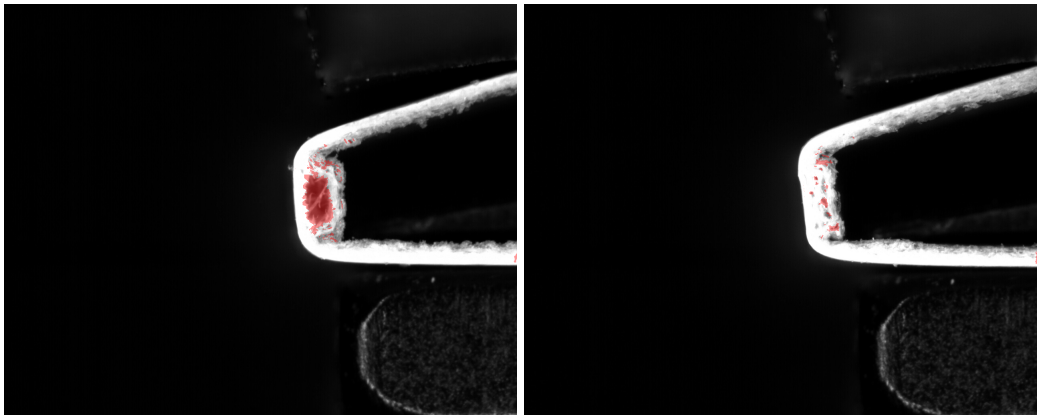

Figure 5.1: *Left*: Paperboard B. *Right*: Paperboard E.

to attention is the fact that paperboard B and E differ so much from each other when it comes to delamination zone area, mostly in crease depths 0.1 mm and 0.2 mm. Looking at figure 5.1 we see that the shape of the bulge differs, one is flat and seems to be more compact will the other one is wider. This is not the only difference of course, we also see that in paperboard B we have a kind of tunnel where the delamination has occured - a clear defined hole probably due to separation of paperboard plies. In paperboard E, the delamination occurs perhaps more between fibers than between plies, not leading to a distinct opening as in paperboard B. This further suggests that in order to classify, say crease depth, one needs to look at more than just delamination zone area. In this case, too, the shape of the bulge would point towards these two having similar crease depth, even though the area is so different.

5.3 Future work

A statistical analysis also incorporating shape parameters such as width, height, and pronunciation of the bulge should be performed. Should the goal be to classify which paperboard has what zone profile, or crease depth, then taking more parameters such as shape into account would be necessary. Furthermore, statistical analysis such as correlating delamination zone size to material parameters like for example bending stiffness and RCS would be interesting.

The proposed method framework could be extended to perform field extraction of materials in other situations as well. A similar study as done here could be made for creasing only, or for creasing and then folding.

Since the method heavily relies on the performance of optical flow estimation methods, it is easy to potentially increase the system performance by simply replacing the optical flow estimator to one that is better. New methods are constantly being developed and a very exciting approach to the optical flow problem is that of self-supervised networks. This would allow for using, say paperboard folding videos, as training data.

Bibliography

[1] Yuling Niu, Shuai Shao, Seungbae Park, and Chin-Li Kao. A novel speckle-free digital image correlation method for in situ warpage characterization. *IEEE Transactions on Components, Packaging and Manufacturing Technology*, 7:276–284, 01 2017.

[2] S. I. Cavlin. The unique convertibility of paperboard. *Packaging Technology and Science*, 1(2):77–92, 1988.

[3] Niclas Stenberg. *On the out-of-plane mechanical behaviour of paper materials*. PhD thesis, KTH, Solid Mechanics, 2002. NR 20140805.

[4] S. Nagasawa, Y. Fukuzawa, T. Yamaguchi, S. Tsukatani, and I. Katayama. Effect of crease depth and crease deviation on folding deformation characteristics of coated paperboard. *Journal of Materials Processing Technology*, 140(1):157 – 162, 2003. Proceedings of the 6th Asia Pacific Conference on materials Processing.

[5] Eric Borgqvist. *Continuum modelling of the mechanical response of paper-based materials*. PhD thesis, Lund University, 2016.

[6] DJ Hine. Testing boxboard creasing. *Modern Packaging*, 8:122–128, 1959.

[7] Kristofer Robertsson, Eric Borgqvist, Mathias Wallin, Matti Ristinmaa, Johan Tryding, Andrea Giampieri, and Umberto Perego. Efficient and accurate simulation of the packaging forming process. *Packaging Technology and Science*, 31(8):557–566, 8 2018.

[8] G.A. Holzapfel. *Nonlinear Solid Mechanics: A Continuum Approach for Engineering*. Wiley, 2000.

[9] Niels Saabye Ottosen and Hans Petersson. *Introduction to the Finite Element Method*. Prentice-Hall, 1992.

[10] S. S. Beauchemin and J. L. Barron. The computation of optical flow. *ACM Comput. Surv.*, 27(3):433–466, September 1995.

[11] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *In IJCAI81*, pages 674–679, 1981.

[12] Jean yves Bouguet. Pyramidal implementation of the lucas kanade feature tracker. *Intel Corporation, Microprocessor Research Labs*, 2000.

[13] Gunnar Farnebäck. Two-frame motion estimation based on polynomial expansion. In *Proceedings of the 13th Scandinavian Conference on Image Analysis*, SCIA'03, pages 363–370, Berlin, Heidelberg, 2003. Springer-Verlag.

[14] Philipp Fischer, Alexey Dosovitskiy, Eddy Ilg, Philip Häusser, Caner Hazirbas, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. *CoRR*, abs/1504.06852, 2015.

[15] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In A. Fitzgibbon et al. (Eds.), editor, *European Conf. on Computer Vision (ECCV)*, Part IV, LNCS 7577, pages 611–625. Springer-Verlag, October 2012.

[16] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. Liteflownet: A lightweight convolutional neural network for optical flow estimation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[17] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. *CoRR*, abs/1612.01925, 2016.

[18] Dassault Systemes Simulia. Abaqus 6.12 documentation. *Providence, Rhode Island, US*, 2012.

[19] Simon Niklaus. A reimplementation of LiteFlowNet using PyTorch. `https://github.com/sniklaus/pytorch-liteflownet`, 2019.

[20] Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE TRANSACTIONS ON IMAGE PROCESSING*, 13(4):600–612, 2004.

[21] Per Erik Austrell, Ola Dahlblom, Jonas Lindemann, Anders Olsson, Karl-Gunnar Olsson, Kent Persson, Hans Petersson, Matti Ristinmaa, Göran Sandberg, and Per-Anders Wernberg. *CALFEM - A finite element toolbox, version 3.4*. Studentlitteratur AB, 2004.

[22] Ken Perlin. Improving noise. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '02, pages 681–682, New York, NY, USA, 2002. ACM.

[23] D. Min, S. Choi, J. Lu, B. Ham, K. Sohn, and M. N. Do. Fast global image smoothing based on weighted least squares. *IEEE Transactions on Image Processing*, 23(12):5638–5653, Dec 2014.

[24] Berthold K. P. Horn and Brian G. Schunck. Determining optical flow. *ARTIFICAL INTELLIGENCE*, 17:185–203, 1981.

Appendices

Appendix A
Dataset

There are 36 sets in total and in each set there should be 4 samples, which is to say that the same setup has been used a number of times for each set. Not all sets have 4 samples due to the fact that in some of them, quality was insufficient or that there was some problem with the specific sample. This results in a total of 127 samples, of which there are video recordings of all 127.

6 different board types have been tested and they all have the same stiffness. The boards have been creased in CD with 0, 0.1, and 0.2 mm crease depth and each test piece is 105 by 150 mm. Tabular A.1 gives a specification of the dataset.

Set	Board	Crease depth (mm)	# Samples
1	A	0	3
2	E	0.1	4
3	D	0	4
4	D	0.2	4
5	C	0.2	4
6	D	0.1	3
7	F	0.1	4
8	E	0.1	4
9	C	0	3
10	B	0	4
11	A	0.1	4
12	A	0	4
13	E	0	3
14	F	0	4
15	C	0	1
16	A	0.1	0
17	F	0	2
18	F	0.2	4
19	B	0	4
20	D	0	4
21	F	0.1	4
22	D	0.2	3
23	C	0.1	4
24	B	0.2	4
25	A	0.2	4
26	E	0	4
27	F	0.2	4
28	B	0.1	3
29	A	0.2	4
30	B	0.2	4
31	B	0.1	4
32	C	0.1	4
33	D	0.1	3
34	E	0.2	3
35	C	0.2	4
36	E	0.2	4

Table A.1: Specification of dataset studied.