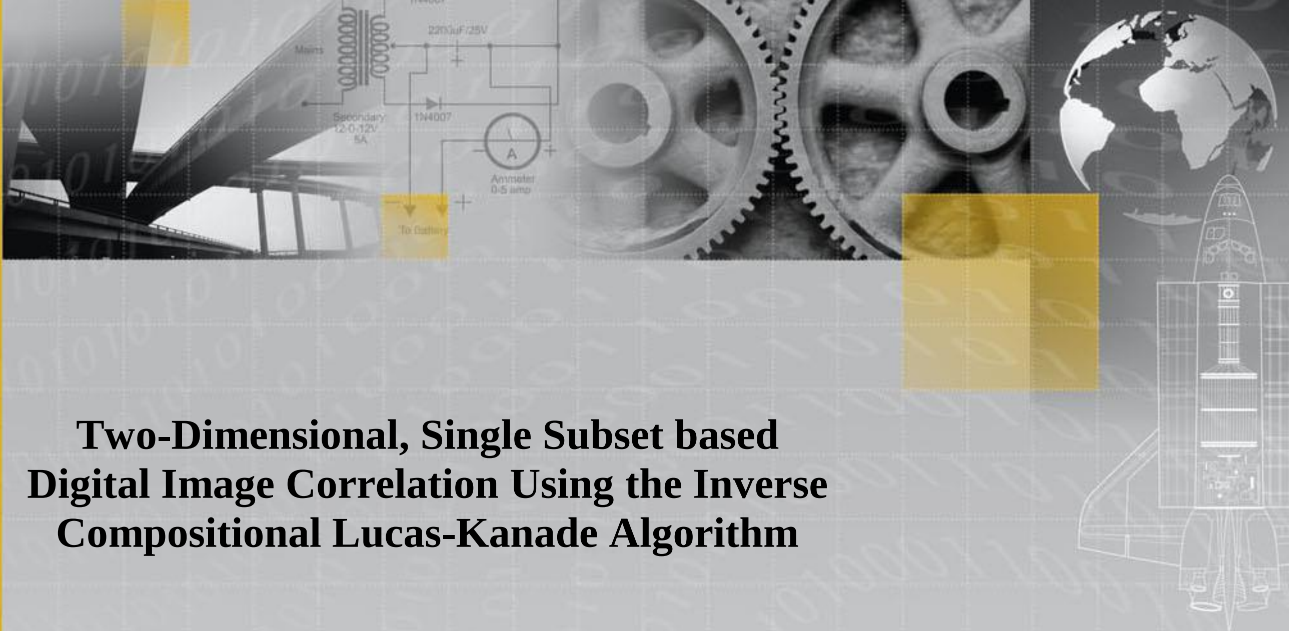
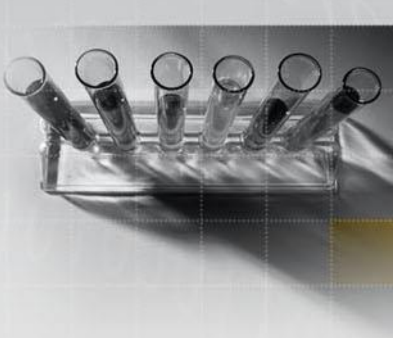


Two-Dimensional, Single Subset based Digital Image Correlation Using the Inverse Compositional Lucas-Kanade Algorithm

Devan Atkinson
17732913



2017

FAKULTEIT INGENIEURSWESE
FACULTY OF ENGINEERING

UNIVERSITEIT
STELLENBOSCH
UNIVERSITY

Department of Mechanical and Mechatronic Engineering
Stellenbosch University

Declaration

I know that plagiarism is wrong.

Plagiarism is to use another's work (even if it is summarised, translated or rephrased) and pretend that it is one's own.

This assignment is my own work.

Each contribution to and quotation (e.g. "cut and paste") in this assignment from the work(s) of other people has been explicitly attributed, and has been cited and referenced. In addition to being explicitly attributed, all quotations are enclosed in inverted commas, and long quotations are additionally in indented paragraphs.

I have not allowed, and will not allow, anyone to use my work (in paper, graphics, electronic, verbal or any other format) with the intention of passing it off as his/her own work.

I know that a mark of zero may be awarded to assignments with plagiarism and also that no opportunity be given to submit an improved assignment.

I know that students involved in plagiarism will be reported to the Registrar and/or the Central Disciplinary Committee.

Name: Devan Atkinson

Student no: 1773 2913

Signature:

Date: 10/10/2017

Contents

List of Figures

Chapter 1

Introduction

Digital Image Correlation (DIC) has become a popular tool in the field of material science to measure the deformations and displacements of test specimens as they are loaded. In order to gain a better understanding of the DIC process this project aims to create an open framework using Matlab that is capable of performing the fundamental elements of DIC. This includes the correlation process which determines the optical flow between images, the calibration process which relates this optical flow between images to full field displacement data given in the real world and post-processing this displacement data to obtain full field strain data.

1.1 Project background

Material science is a field of study that focuses mainly on quantifying the characteristics of materials. These quantified characteristics are referred to as material properties and they are used to predict the behaviour of a material when under certain boundary conditions. The Material Engineering group at Stellenbosch University focuses specifically on mechanical property determination of materials.

These mechanical properties are used in constitutive equations to approximate the response of a material, in terms of deformation, to external stimuli, such as a force, where the material property indicates the degree of response of the material. Thus the constitutive equations mathematically define the material properties such that they can be quantified.

This is important since these constitutive equations and material properties can be used to predict the behaviour of a component when subjected to a external force that the component is expected to withstand during its use. This predicted behaviour can then be used to determine whether the component is susceptible to failure during its intended use case. However the accuracy of this predicted behaviour is dependent on the accuracy of the material properties used.

As such ASTM standards have been developed to accurately and reliably determine material properties. These ASTM standards rely upon using conventional displacement measurement devices which typically determine displacement as the change in distance between two points on the specimen. For example the ASTM Standard E8 [1] relies upon using an extensometer to measure elongation and the ASTM Standard E399 [2] requires a displacement gauge to measure the crack-mouth opening displacement. Therefore the ASTM standards place strict requirements on the specimen geometry for two reasons.

First to ensure that the deformation experienced by the specimen over the portion of the specimen that lies between the measurement points is directly dependent on the material property to be determined. Secondly to either mitigate the effects that other material properties play in this deformation or have them influence the deformation in a way that can be reliably accounted for. This is done so that a specialised constitutive equation which is calibrated to a specific specimen, in terms of how the displacement is measured, and only accounts for one material property can be used to quantify the material property. Consequently a different specimen is required in order to accurately determine each material property.

However the use of optical full-field measurement techniques to determine displacement across the full-field of the specimen have been successfully applied within the field of Material Science [13, 14]. DIC is one such technique which determines displacements experienced across the surface of the specimen from images taken of the specimen as it deforms.

Subsequently, because this technique provides displacements across the whole surface of the specimen more complex constitutive equations, that include the effects of multiple material properties, can be used. This allows more than one material property to be determined during a single experiment as investigated by Huchzermeyer [17]. This also relaxes the rigid requirements that were placed on the specimen geometry when using conventional measurement techniques. Now the geometry need only ensure that the material properties that are to be determined play a significant enough role in the deformation of the specimen.

Additionally since DIC is a non-contact optical measurement technique the specimen can be exposed harsh environments while measurements can still be taken as long as there is a clear view of the specimen. For example DIC can be used to measure the creep of a material that is exposed to high temperatures in an oven while a camera captures images of the specimen through a window in the oven [29].

1.2 Project motivation

Subset based DIC breaks up the reference image into clusters of pixels, called subsets, and then determines the deformation and displacement experienced by each subset such that it becomes identical to a corresponding subset in the deformed image. As such subset based DIC basically allows you to place thousands of virtual strain gauge rosettes, which measure both deformation and displacement, over a surface of your specimen.

However these virtual strain gauge rosettes operate differently from regular ones. A regular strain gauge rosette consists of several strain gauges arranged in a certain orientation relative to one another. Although all the strain gauges of the rosette experience the same deformation each measures a strain in a certain direction. These directional strains are then used to calculate the two dimensional strain tenor experienced by the material at a specific location on the specimen.

In contrast the virtual strain gauge rosettes don't use multiple strain gauges to independently measure strains in multiple directions. Instead the virtual strain gauge rosette consists of a single, complex strain gauge which measures the full two dimensional strain tensor and the displacement it experiences. This is done by first defining a function, called the warp function, which describes how the subset is capable of deforming. The parameters of this warp function, the warp function parameters (WFPs), are then optimised such that when the reference subset is deformed and displaced according to these parameters it matches the corresponding subset in the deformed image.

Two of these WFPs are directly related to the displacement experienced by the subset between images. As such once the WFPs of all the subsets have been optimised the displacement field experienced by the specimen can be extracted. This displacement field is then differentiated in order to obtain the strain fields of the specimen.

It is important to note that although the warp function defines how the subsets can deform and displace; the WFPs which quantify its deformation are not used to calculate the strain fields. Only the parameters that quantify displacement are used. The purpose of the deformation parameters is to allow the subset to deform such that the displacement parameters determined for a subset are not biased by the deformation experienced by that subset. Therefore the allowable deformation defined by the warp function needs to be appropriate for the deformation experienced by the specimen in order for the displacement results to be reliable.

Most commercial software packages have a fixed set of warp functions which can be used; the most popular being the first and second order warp functions. These warp functions are sufficient to account for general material deformation such that the displacement results are reliable. However as the displacement gradients becomes more severe these warp functions require a reduction in the subset size in order to avoid errors in approximating the underlying deformations.

Despite this, it has been shown by Pan et al. [23] that a reduction in subset size leads to a decrease in accuracy of the calculated displacements. As such for severe displacement gradients it is better to increase the order of the warp function than decrease the subset size. However commercial software seldom offers warp functions of higher order than second order.

The DIC framework proposed in the project has been developed in a modular way such that a wide range of warp functions can be defined by the user. Thus warp functions of higher order than second order can be defined for severe displacement gradients in order to more reliably capture the underlying displacements. However the flexibility of the warp function in this DIC framework offers additional advantages.

Although it is true that defining a warp function which does not account for deformations that the subset undergoes results in biasing the displacement results, allowing for deformations in the warp function that do not take place can also bias the displacement results. For example a standard dogbone specimen experiences only normal strain with no shear strain. As such the WFPs related to shear will have values almost equal to zero. However because these WFPs are not exactly zero they cause noise in the displacement results which is undesirable.

Thus by removing the shear component from the warp function this bias can be eliminated. Allowing the user to define the warp function enables them to use a priori knowledge to define more appropriate warp functions in order to avoid biasing the displacement results.

Therefore the user can define a warp function which only accounts for normal strain in the deformation in order to obtain more accurate displacement results.

what is the need for custom Wf directly map what look for.

Most commercial software packages have a fixed set of warp functions which can be used; the most popular being the first order affine warp function. These warp functions are sufficient to account for general material deformation such that the displacement results are reliable. However when the deformation of the specimen becomes complex, such as near a stress concentration like a crack tip, these warp functions cannot account for the deformation properly and the displacement results become biased. As such it has become common practice to avoid displacement results near severe stress concentrations such as a crack tip.

In contrast the DIC framework proposed in the project offers full control over the warp function such that users can define their own warp function.

boundaries of bad displacement data that cannot be used new method get multiple material prop from single specimen and drive to reduce size of this specimen as such removing boundaries of bad displacement is important

1.3 Objectives

The aim of this project is to create a DIC algorithm which provides the user with more control over setting up the correlation process. In order to do so the following key objectives need to be achieved.

- Complete a comprehensive literature review on gradient-descent, subset-based DIC.
- Use literature review to develop a basic flow diagram of the steps involved in the correlation process.
- Develop basic Newton-Raphson and Lucas-Kanade DIC algorithms that are capable of determining full-field displacements across the surface of a specimen from images taken of the specimen.
- Modify the Lucas-Kanade algorithm so that is it more modular allowing the user to set up the correlation problem.
- Validate this modular algorithm by testing how well it correlates synthetic images of known displacements. Additionally use this to show how setting up the correlation problem for specific displacement fields can be beneficial.
- Compare the performance of the algorithm to commercial software by correlating experimental image sets and comparing the calculated displacements.

1.4 Contents

This report starts off by providing background information on DIC. This includes the basic operations of a camera, how a 3D scene is projected onto the imaging sensor of a camera, the basic calibration process and finally the basic elements of the correlation algorithm. Thereafter the optimisations algorithms of the correlation process are discussed in detail. How these optimisation algorithms have been implemented wihtin MATLAB are then explained.

Finally the presented DIC framework is validated. Two methods of validation, namely synthetic and experimental, are used to validate the proposed framework. Synthetic validation is used to validate the correlation component of the framework whereas experimental validation validates the overall framework.

1. Introduction

    (a) Project background

   (b) Project motivation
   (c) Objectives

2. Literature review

   (a) Digital cameras
   (b) Camera optics
   (c) World to sensor coordinates (give other coordinate systems upon which this is based in the appendices)
   (d) Distortion (give ones that are accounted for while others in appendices)
   (e) Calibration
      i. Inverse problem
      ii. Calibration plate
      iii. Homography
      iv. Estimating homography using direct linear transformation
      v. Absolute conic
      vi. Constraints on the intrinsic parameters
      vii. Intrinsic parameters and the absolute conic
      viii. Distortion in calibration
      ix. Non-linear optimisation
   (f) Correlation
      i. Correspondence problem and speckle patterns
      ii. Correlation process (give flow diagram of correlation process and elaborate on the elements involved)
         A. Correlation criteria
         B. Warp function
         C. Interpolation
         D. Optimization

3. Correlation algorithms

   (a) Newton-raphson
      i. Code explanation
   (b) Lucas-Kanade
      i. Code explanation
   (c) Phase shift correlation
      i. Code explanation

(d) Subset splitting

i. Code explanation

4. Synthetic validation

(a) Generating synthetic images

i. Speckle function

ii. Deformed images

iii. Code explanation

(b) Displacement fields (used to create synthetic images)

i. Plate with hole

ii. Plate with crack

(c) Results

5. Experimental validation

(a) Specimens

i. Tension specimen with hole

ii. CT specimen

iii. Arcan specimen

(b) Results

6. Conclusion

Explain code in a section after the mathematics are presented so it easier to illustrate what is going on (link maths to code)

Chapter 2

Literature review

This chapter outlines the theory that is relevant for Digital Image Correlation. First the digital camera is discussed in order to understand how it captures light and stores it as data. Thereafter the optical system, made of a system of lenses and apertures, which focuses the light for the digital camera is reviewed since it has a significant influence on how a three dimensional scene is converted into two dimensional data. Then the camera model that is used to mathematically relate three dimensional coordinates in the real world to two dimensional pixels in the image is presented. Lastly the distortions that are caused by the slight imperfections of lenses are explained.

2.1 Digital cameras

Cameras at the basic level rely upon using an aperture and a lens to focus light rays that originate from objects onto a plane within the camera called the sensor plane. At the sensor plane exist a Charge-Coupled Device (CCD) that consists of a matrix of light sensors. Each light sensor converts the light incident upon its surface into an electrical charge through the photoelectric effect. The charge is proportional to the light intensity. Then each light sensor's voltage is read by an analogue-to-digital converter which measures the voltage and assigns a digital value to it. These digital values are then stored at the corresponding position in a matrix which forms the digital image.

2.2 Homogeneous coordinates

It is common knowledge that any object's shape can be fully defined using distances and angles in 3D Euclidean space. However when an image is taken of this object these distances and angles become distorted. For example railway tracks consist of two beams that remain parallel to one another at a set distance apart in Euclidean space but in an image of the railway track (projective space) these beams appear to get closer and closer to one another as seen in figure 2.1.

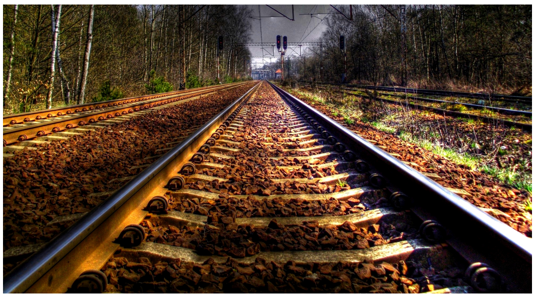


Figure 2.1: How projective space distorts parallelism present in Euclidean space

Therefore the parallelism between the beams in Euclidean space is distorted in projective space. This phenomenon is referred to as perspective, where an object appears smaller the further away it is, and occurs as a result of reducing 3D information to a 2D image. According to this phenomenon a coordinate in 3D Euclidean space is mathematically related to a 2D coordinate in projective space through a scaling variable.

Homogeneous coordinates are simply Euclidean coordinates with this scaling variable added as an additional element in the coordinate vector. Homogeneous coordinates can be converted back to euclidean coordinates by scaling the vector by the scaling variable and then removing the scaling variable from the vector. In other words the homogeneous vector is multiplied by the inverse of the scaling variable and then the scaling variable is dropped from the vector.

Homogeneous coordinates aren't only useful for converting a coordinate from Euclidean space to projective space. They also allow translations and rotations to be applied to a coordinate vector using matrix multiplication.

2.3 Camera optics

For an image of an object to be in focus the light incident upon each light sensor should originate from one point on the object's surface. However light is reflected by objects in many directions and so a means of focusing the light is required. This is accomplished through using lenses and an aperture. Throughout this project thin lenses are assumed. A thin lens is one in which its thickness is negligible in comparison with its focal length or radius of curvature [27]. Additionally the paraxial approximation is assumed which states that light rays passing through the lens do so with a small angle to the lens's optical axis and pass through the lens close to the optical axis. This leads to the small angle approximation.

$$\sin(\theta) \approx \tan(\theta) \approx \theta \tag{2.1}$$

Lenses are usually disk shaped pieces of glass with two convex surfaces. The convex surfaces are designed to bend light towards the optical axis with the degree of bending increasing with the distance from the light to optical axis at the lens mid plane. Thus diffuse light emanating from a point M $[x, y, z]^T$ on an object will pass through the lens and the light rays will converge at a point called the ideal image point M' $[x', y', z']^T$. Thereafter the light rays diverge again to points M" $[x'', y'', z'']^T$ as shown in Figure 2.2. If the sensor plane is coincident with the ideal image points of the light rays the image that forms on the sensor is inverted due to the way the lens bends the light. As a result an inverted coordinate system is used for the sensor plane.

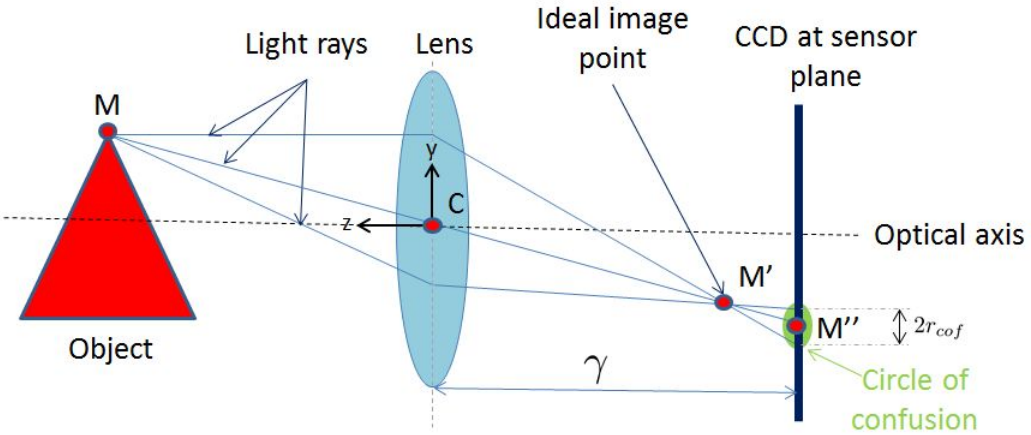


Figure 2.2: Illustration of how light rays are manipulated by the lens

The thin lens equation can be stated as

$$\frac{1}{|CM|} + \frac{1}{CM'} = \frac{1}{\bar{f}} \tag{2.2}$$

where $\bar{f}$ is the focal length; an inherent property of the lens. Additionally through similarity of triangles we have

$$\frac{y}{CM} = \frac{-y'}{CM'} \tag{2.3}$$

$$\frac{x}{CM} = \frac{-x'}{CM'} \tag{2.4}$$

$$\frac{z}{CM} = \frac{-z'}{CM'}. \tag{2.5}$$

Combining these a series of equations for the ideal image point can be obtained.

$$\begin{aligned} x' &= \frac{-\bar{f}x}{z-\bar{f}} \\ y' &= \frac{-\bar{f}y}{z-\bar{f}} \\ z' &= \frac{-\bar{f}z}{z-\bar{f}} \end{aligned} \tag{2.6}$$

2.3.1 Circle of confusion

It is impossible to align the sensor plane perfectly with the ideal image points and so the divergence of the light rays after the ideal image point causes the light rays to illuminate a circular area on the sensor plane. This blurred region is known as the circle of confusion. If this circular area is larger than the area spanned by an element of the sensor then this will result in blurring in the image as the light spills over multiple sensors. This blur could be eliminated by having the sensor plane the same distance from the lens as the the ideal image point but different points on the object will have different ideal image points.

The light rays that make up the outer perimeter of the circle of confusion are the rays that pass through the lens on the outer edges. Using these rays it is possible to determine the radius of the circle of confusion. Taking r to be the radius of the lens these outer light rays pass through the midplane of the lens along the perimeter of a circle described by $[r\cos\beta, r\sin\beta, 0]^T$ where $0 \leq \beta \leq 2\pi$. Here β is the angle subtended by the radius and the x axis that lies in plane with the lens. Using trigonometry the x and y components of these light rays can be related as

$$\frac{x' - r\cos\beta}{z'} = \frac{x' - x''}{\gamma + z'} \tag{2.7}$$

$$\frac{y' - r\sin\beta}{z'} = \frac{y' - y''}{\gamma + z'} \tag{2.8}$$

The distance between the midplane of the lens and the sensor plane is given by γ. Using these and Equation 2.6 an expression for the perimeter of the circle of confusion can be found.

$$\begin{bmatrix} x'' \\ y'' \\ z'' \end{bmatrix} = \begin{bmatrix} r\cos\beta + r\cos\beta\left(1 + \frac{\gamma(\bar{f}-z)}{\bar{f}z}\right) \\ r\sin\beta + r\sin\beta\left(1 + \frac{\gamma(\bar{f}-z)}{\bar{f}z}\right) \\ 0 \end{bmatrix} + \begin{bmatrix} -\frac{\gamma x}{z} \\ -\frac{\gamma y}{z} \\ -\gamma \end{bmatrix} \tag{2.9}$$

The radius of the circle of confusion can be determine by taking the difference in the y components of y'' for $\beta = 90°$ and $\beta = 270°$.

$$2r_{cof} = \left[r\sin 90° \left(2 + \frac{\gamma\left(\bar{f} - z\right)}{\bar{f}z}\right)\right] - \left[r\sin 270° \left(2 + \frac{\gamma\left(\bar{f} - z\right)}{\bar{f}z}\right)\right] \quad (2.10)$$

$$r_{cof} = r\left(1 + \frac{\gamma\left(\bar{f} - z\right)}{\bar{f}z}\right) \quad (2.11)$$

Another way that the blur can be improved is by using an aperture. An aperture is at a basic level an opaque diaphragm with a hole in it which serves to reduce the amount of light that is incident upon the sensor. Thus by blocking off the light that passed through the outer edges of the lens the aperture reduces the size of the circle of confusion by effectively reducing the radius of the lens. The aperture can be place in front or behind the lens.

2.3.2 Depth of field

Depth of field is defined as the distance ahead and behind the object that is in focus. For a particular camera system the ideal image point of an object will fall upon the sensor plane if the object is at an ideal focal length from the lens. If the object is any closer or further from the lens it will cause a circle of confusion on the sensor plane as opposed to a point. If the circle of confusion is small enough it will not spill significant light over multiple sensors which will result in a clear image.

However if the circle of confusion is large enough it will cause blurring in the image. The largest circle of confusion which still results in a sharp image is referred to as the acceptable circle of confusion [27]. Thus the depth of field can be considered as the distance along the z-axis ahead or behind the ideal focus length which results in an acceptable circle of confusion.

The largest degree of freedom for a specific camera system occurs when the object is at the hyperfocal length, H, from the camera. The hyperfocal length is defined as the closest an object can be to the camera such that the depth of field extends to infinity behind the object. In this situation the depth of field starts at a distance of $\frac{H}{2}$. The hyperfocal length can be approximated as

$$H \simeq \frac{\bar{f}^2}{2Nr_{cof}} \quad (2.12)$$

Here N is given by $\frac{\bar{f}}{D_p}$, where D_p is the diameter of the entrance pupil for the camera system. Letting s be the distance from the camera to the object such that the camera is ideally focused at a distance s. The distance from the camera to the near limit of the depth of field, D_N, and the distance from the camera to the far limit of the depth of field, D_F, can be approximated.

$$D_N \simeq \frac{Hs}{H+s} \tag{2.13}$$

$$D_F \simeq \frac{Hs}{H-s} \tag{2.14}$$

These approximations assume that the object distance is large compared to the lens focal length. The depth of field can then be determined.

$$DOF = D_F - D_N = \frac{2Hs^2}{H^2 - s^2} \tag{2.15}$$

Combining Equation 2.12 and 2.15

$$DOF = \frac{4Nr_{cof}\bar{f}^2 s^2}{\bar{f}^4 - 4N^2 r_{cof}^2 s^2}. \tag{2.16}$$

Thus it is clear that the depth of field can be controlled by altering the focal length of the lens. This can be done by either changing the size of the aperture D_p or by changing the distance between the camera and the object.

2.3.3 Field of view

The field of view is the extent of the world that the camera is capable of capturing in an image. It is quantified as the largest angle that a light ray, that is incident upon the sensor, makes with the optical axis. This angle is referred to as the angle-of-view.

viewing frustum

Using the pinhole camera model with a distance of L between the sensor and the lens and taking the lens height to be d a relation for the angle-of-view, α, can be derived using trigonometry.

need to illustrate these dimensions in an image

$$\tan\left(\frac{\alpha}{2}\right) = \frac{d}{2L} \tag{2.17}$$

$$\alpha = 2\arctan\left(\frac{d}{2L}\right) \tag{2.18}$$

However for the best picture quality the distance between the sensor and the lens should be equal to the focal length f.

$$\alpha = 2\arctan\left(\frac{d}{2f}\right) \tag{2.19}$$

2.3.4 Front image plane model

A new imaging model that is often preferred for computer vision applications can be obtained by translating the sensor plane a distance of 2γ along the optical axis. In this case the sensor plane is in front of the lens. Treating the sensor coordinates M'' of an object as the intersection of the light ray with the sensor plane; Equation 2.22 remains valid for this configuration.

This imaging model is advantageous in that the sensor plane coordinates are no longer inverted. This means that the scene being imaged is also not inverted.

the perspective projection does not involve negative values now

2.4 Coordinate systems

There are four coordinate systems involved in converting a coordinate on an object in the real world to a corresponding coordinate in an image of the object.

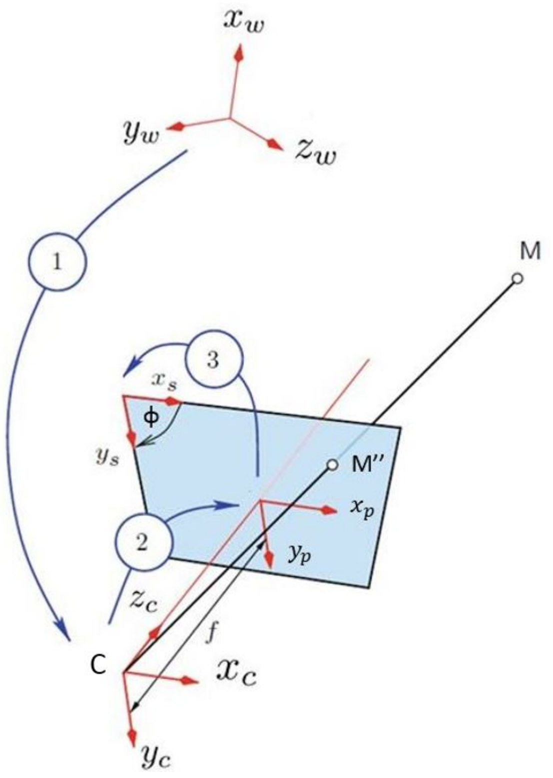


Figure 2.3: The conversion between coordinate systems that occurs when an image is taken [27]

These coordinate systems are the world coordinate system represented by subscript w, the camera coordinate system represented by subscript c, the image plane coordinate system represented by subscript p and the sensor coordinate system represented by subscript s. This section presents these four coordinate systems and the mathematical relations used to convert coordinates between them.

2.4.1 World to camera coordinate system

The world coordinate system is simply the coordinate system that is used to classify the position of objects in the real world,(x_w, y_w, z_w). Its orientation and origin are arbitrary and it is usually classified according to the calibration plate used. The camera coordinate system is fixed according to the cameras position and orientation where the z-axis is the optical axis of the camera. Camera coordinates are represented as (x_c, y_c, z_c).

Can i say calibration plate without defining it yet

Converting from the world coordinate system to the camera coordinate system involves rigid transformations of translation T and rotation R.

$$\boldsymbol{X}_c = \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{T} \\ \boldsymbol{0} & 1 \end{bmatrix} \boldsymbol{X}_w. \tag{2.20}$$

The parameters in $\boldsymbol{R}$ and $\boldsymbol{T}$ are referred to as extrinsic parameters since they are not dependent on the camera system's hardware but rather the camera's position and orientation in the world coordinate system.

change T and R to lowercase

2.4.2 Camera and image plane coordinates

The image plane coordinate system is an orthogonal coordinate system that has its x-y plane coincident with the plane of the CCD sensor array. Its z-axis is also coincident with the z-axis of the camera coordinate system. Although this coordinate system has three dimensions its coordinates all lie on the x-y plane and so it can be treated as a 2D coordinate system.

improve this

Conversion of a coordinate from the camera coordinate system to the image plane coordinate system involves perspective projection. This relation can be derived from Equation 2.9 by eliminating the circle of confusion component and substituting the appropriate dimensions. Additionally when the camera system is set up properly z' will be equal to γ so that the sensor plane is approximately coincident with the ideal image points.

$$\begin{bmatrix} x'_c \\ y'_c \\ z'_c \end{bmatrix} = \begin{bmatrix} \frac{\gamma}{z_c} & 0 & 0 \\ 0 & \frac{\gamma}{z_c} & 0 \\ 0 & 0 & \gamma \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ 1 \end{bmatrix} \tag{2.21}$$

Here (x'_c, y'_c, z'_c) are the camera coordinates projected onto the image plane but stipulated within the camera coordinate system. Note that the since the front image plane model is used the dimensions are now non-negative. The issue with this equation is the dependence on z_c in the matrix. This is fixed by using a homogeneous form of Equation 2.21.

$$\alpha \begin{bmatrix} x'_c \\ y'_c \\ z'_c \\ 1 \end{bmatrix} = \begin{bmatrix} \gamma & 0 & 0 & 0 \\ 0 & \gamma & 0 & 0 \\ 0 & 0 & \gamma & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} \tag{2.22}$$

This equation is referred to as perspective projection [27]. Here α is the scaling variable of the homogeneous coordinates. It is divided out of the homogeneous coordinate vector so that the coordinate is correctly scaled on the image plane. The value of α is equal to z_c and z'_c is equal to γ such that this coordinate lies on the image plane.

make sure this is correct

Converting this projected coordinate to the image plane coordinate system is achieved by dropping the z dimension to obtain

$$\boldsymbol{X}_p = \begin{bmatrix} x_p \\ y_p \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x'_c \\ y'_c \\ z'_c \\ 1 \end{bmatrix} = \begin{bmatrix} \gamma & 0 & 0 & 0 \\ 0 & \gamma & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix} \tag{2.23}$$

2.4.3 Image plane and sensor coordinates

At this point the locations of where the light rays originating from the object will intersect the image plane are known. Now it is necessary to mathematically represent how the sensor would interpret these light rays incident upon the sensor into the form of a matrix of pixels. In order to do this a relation between the position of a point within a coordinate system and the pixel within an image is necessary. Additionally the sensor array is not guaranteed to the orthogonal and so a skewed coordinate system must be taken into account.

The transformation from the image plane coordinate system to a temporary skewed coordinate system with an angle of ϕ between the two axes can be represented by

$$\begin{bmatrix} x_{temp} \\ y_{temp} \end{bmatrix} = \begin{bmatrix} 1 & -\cot\phi \\ 0 & \frac{1}{\sin\phi} \end{bmatrix} \begin{bmatrix} x_p \\ y_p \end{bmatrix}. \tag{2.24}$$

It is assumed that the two principle directions in the sensor coordinate system have different scale factors, S_x and S_y, which have units of pixels per unit length. Additionally the sensor coordinate system has its origin at one of the corners of the image plane such that all coordinates that fall on the image plane are positive. This is because these coordinates represent positions within a matrix. As such the coordinates need to be translated by $\hat{c}_x$ and $\hat{c}_y$ to account for the origin of the sensor coordinate system.

Note that these translations are given in terms of the image plane coordinate system. Applying these to the coordinates calculated in equation 2.24 results in the sensor coordinates below.

$$\begin{bmatrix} x_s \\ y_s \end{bmatrix} = \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix} \begin{bmatrix} x_{temp} \\ y_{temp} \end{bmatrix} - \begin{bmatrix} S_x \hat{c}_x - S_x \hat{c}_y \cot\phi \\ \frac{S_y \hat{c}_y}{\sin\phi} \end{bmatrix} = \begin{bmatrix} S_x & -S_x \cot\phi \\ 0 & \frac{S_y}{\sin\phi} \end{bmatrix} \begin{bmatrix} x_p \\ y_p \end{bmatrix} - \begin{bmatrix} S_x \hat{c}_x - S_x \hat{c}_y \cot\phi \\ \frac{S_y \hat{c}_y}{\sin\phi} \end{bmatrix} \tag{2.25}$$

This can be rewritten in homogeneous form so that it is consistent with equation 2.23.

$$\boldsymbol{X}_s = \begin{bmatrix} x_s \\ y_s \\ 1 \end{bmatrix} = \begin{bmatrix} S_x & -S_x \cot\phi & -S_x \left(\hat{c}_x - \hat{c}_y \cot\phi\right) \\ 0 & \frac{S_y}{\sin\phi} & -\frac{S_y \hat{c}_y}{\sin\phi} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_p \\ y_p \\ 1 \end{bmatrix} = \boldsymbol{A}\boldsymbol{X}_p \tag{2.26}$$

2.4.4 World to sensor coordinates

talk about these parameters being intrinsic and combine them better

The conversions between coordinate systems described in Equations 2.20, 2.23 and 2.26 can be combined into one conversion from the world coordinate system to the sensor coordinate system.

$$\boldsymbol{X}_s = \alpha \begin{bmatrix} x_s \\ y_s \\ 1 \end{bmatrix} = \begin{bmatrix} S_x & -S_x \cot\phi & -S_x \left(\hat{c}_x - \hat{c}_y \cot\phi\right) \\ 0 & \frac{S_y}{\sin\phi} & -\frac{S_y \hat{c}_y}{\sin\phi} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \gamma & 0 & 0 & 0 \\ 0 & \gamma & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \tag{2.27}$$

This can be simplified by combining the first two matrices.

$$\boldsymbol{X}_s = \alpha \begin{bmatrix} x_s \\ y_s \\ 1 \end{bmatrix} = \begin{bmatrix} \gamma S_x & -\gamma S_x \cot\phi & -S_x \left(\hat{c}_x - \hat{c}_y \cot\phi\right) & 0 \\ 0 & \frac{\gamma S_y}{\sin\phi} & -\frac{S_y \hat{c}_y}{\sin\phi} & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \tag{2.28}$$

Replacing the elements of the first matrix in equation 2.28 with single variables for the purposes of simplicity the equation can be rewritten as

$$\alpha \begin{bmatrix} x_s \\ y_s \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & f_s & c_x & 0 \\ 0 & f_y & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \tag{2.29}$$

$$= \boldsymbol{K}\boldsymbol{V}\boldsymbol{X}_w. \tag{2.30}$$

Here the parameters relating the world coordinate system to the sensor coordinate system are separated into two matrices. The first matrix $\boldsymbol{K}$ contains the intrinsic parameters which are fixed for a specific camera since they are dependent on the hardware that makes up the camera system. The second matrix $\boldsymbol{V}$ contains the extrinsic parameters that explain how the camera is orientated with respect to the calibration plate which defines the world coordinate system. Thus extrinsic parameters change when the camera's position and orientation relative to the world coordinate system changes.

2.5 Distortion

Distortion refers to a collection of phenomena that cause the actual image to differ from that of the idealised image expected from the pinhole camera model. This happens because lenses cannot be manufactured and assembled perfectly and so some defects and misalignments exist in the imaging system. Therefore in order to calculate accurate displacement information the distortions must be accounted for.

Although there are many types of distortion, it has been found that the difference between the actual image and the idealised one can be well accounted for by only considering radial distortion [28,30]. As such this section will focus on this type of distortion only.

2.5.1 Radial

Radial distortion is caused by the lens having different magnification levels based on the angle of the light rays to the optical axis. As a result the image can experience a decrease in magnification with increasing distance from the optical axis (barrel distortion) or the image can experience increasing magnification with increasing distance from the optical axis (pincushion distortion). This distortion is symmetric with respect to the optical axis.

Radial distorion is mathematically accounted for in the image plane coordinate system. An undistorted coordinate, (x_p, y_p), is distorted to coordinate $(\hat{x}_p, \hat{y}_p)$ according to the relation

$$\begin{bmatrix} \hat{x}_p \\ \hat{y}_p \end{bmatrix} = (1 + \kappa_1 r^2 + \kappa_2 r^4) \begin{bmatrix} x_p \\ y_p \end{bmatrix} \tag{2.31}$$

$$\text{where} \quad r = \sqrt{x_p^2 + y_p^2} \tag{2.32}$$

Here κ_1 and κ_2 are the radial distortion parameters which describe the severity of the distortion. For barrel distortion these parameters are possitive and for pincushion distortion these parameters are negative. (Note: I still need to add an image that shows what radial distortion looks like)

possibly need an image and explanation of e_r

2.6 Calibration

Calibration is a necessary process that must be completed in order to extract metric information from images. The calibration process solves for the parameters $\boldsymbol{K}$ and $\boldsymbol{V}$ in Equation 2.30 and parameters κ_1, κ_2 and κ_3 in Equation 2.31. Thus it solved for the extrinsic, intrinsic and radial distortion parameters. In doing so the calibration process will have solved the camera model that predicts how a coordinate on an object, in three dimensional space, is related to the corresponding pixel in the image captured of the object.

Multiple methods of calibration exist however calibration using a calibration plate is used in this project since it is one of the most popular methods and manufacturing a calibration plate is relatively simple and inexpensive.

2.6.1 Inverse problem

Calibration is essentially a method of finding the parameters that allow 3D coordinates in the world coordinate system to be accurately related to 2D coordinates in the image. Thus the inputs, 3D world coordinates, and outputs, 2D image coordinates, needed to be used to solve for the parameters which describe the relationship between the two which constitutes an inverse problem.

Inverse problems are typically hard to solve and calibration is no exception. In order to solve for the calibration parameters more reliably the camera model that relates world points to sensor points is broken down to the simple pinhole camera model initially in order to solve for as few parameters as possible in the beginning. These parameters are solved for using a closed form solution which gives good estimates to the intrinsic and extrinsic parameters. Thereafter once these parameters have been estimated the camera model is made more complex by introducing radial distortion in order to account for imperfections in the lens system of the camera. These radial parameters are first estimated and now with each parameter having a corresponding estimate all the parameters are optimized simultaneously in an iterative manner.

This inverse problem is sensitive to errors in the 3D world coordinates (inputs) and 2D image coordinates (outputs) used. Thus these need to be known to a high degree of accuracy in order to solve for the intrinsic and extrinsic camera parameters reliably. This is achieved by using a calibration plate.

2.6.2 Calibration plate

A calibration plate is an object with a flat surface containing a high-contrast, regular pattern. The pattern is such that it contains definitive, point-like features which can be located to a high degree of accuracy within images taken of it. For example a checker board pattern allows for accurate calculation of the points at the corners of the squares. Thus the coordinates of these point-like features on the calibration plate will be known (inputs) and the coordinates of these point-like features in the image can be determined to a high degree of accuracy (outputs). These definitive, point-like features are hereafter referred to as calibration targets.

Since images inherently contain some level of noise it is best to have an overdetermined system of equations. This is accomplished by taking multiple images of the calibration plate and changing the relative position and orientation between the calibration plate and the camera for each image. This effectively reduces the effects of noise; making the system more robust.

2.6.3 Homography

Homography is a transformation that can be applied to points on a plane to bring it into alignment with another plane. It is used to bring the calibration targets on the calibration plate in the world coordinate system into alignment with their location in the image in the sensor coordinate system. The transformation from world coordinates to sensor coordinates in equation 2.30 is a type of homography.

The calibration plate defines the position and orientation of the world coordinate system. It is customary to do so in such a way that the calibration targets lie on the x-y plane of the world coordinate system. Then the homography, $\boldsymbol{H}$, that relates these calibration targets to their corresponding position in the image is derived using Equation 2.30.

$$\begin{bmatrix} x_s \\ y_s \\ 1 \end{bmatrix} = \alpha \begin{bmatrix} f_x & f_s & c_x & 0 \\ 0 & f_y & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ 0 \\ 1 \end{bmatrix} \tag{2.33}$$

Here $(x_w, y_w, 0)$ represents a calibration target of the calibration plate in the world coordinate system while (x_s, y_s) represents the corresponding projection of the calibration target in the sensor coordinate system. Since the z-component of the world coordinate system is zero this equation can be reduced to

$$\begin{bmatrix} x_s \\ y_s \\ 1 \end{bmatrix} = \alpha \begin{bmatrix} f_x & f_s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & t_1 \\ r_{21} & r_{22} & t_2 \\ r_{31} & r_{32} & t_3 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ 1 \end{bmatrix} \tag{2.34}$$

$$= \alpha \boldsymbol{K} \begin{bmatrix} \boldsymbol{r}_1 & \boldsymbol{r}_2 & \boldsymbol{t} \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ 1 \end{bmatrix} \tag{2.35}$$

$$= \alpha \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ 1 \end{bmatrix} = \alpha \boldsymbol{H} \begin{bmatrix} x_w \\ y_w \\ 1 \end{bmatrix} \tag{2.36}$$

Here $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ are the first and second columns of the rotation matrix. It is clear that the homography matrix contains both intrinsic and extrinsic parameters and thus it is different for each image taken of the calibration plate. Additionally note that the homography matrix is defined up to a scale factor α.

By reducing Equation 2.30 to this form the amount of parameters that need to be solved for has decreased from 17 to 9. This helps to reduce the difficulty in solving the inverse problem.

2.6.4 Estimating homography with direct linear transformation

The homography of Equation 2.36 can be estimated using direct linear transformation (DLT) [12]. Equation 2.36 can be written out as

$$x_s = \alpha \left(h_{11} x_w + h_{12} y_w + h_{13} \right) \tag{2.37}$$

$$y_s = \alpha \left(h_{21} x_w + h_{22} y_w + h_{23} \right) \tag{2.38}$$

$$1 = \alpha \left(h_{31} x_w + h_{32} y_w + h_{33} \right). \tag{2.39}$$

The scale factor, α, can be eliminated by dividing equations 2.37 and 2.38 by 2.39 to get

$$x_s \left(h_{31} x_w + h_{32} y_w + h_{33} \right) = \left(h_{11} x_w + h_{12} y_w + h_{13} \right) \tag{2.40}$$

$$y_s \left(h_{31} x_w + h_{32} y_w + h_{33} \right) = \left(h_{21} x_w + h_{22} y_w + h_{23} \right) \tag{2.41}$$

These equations then reduce to

$$-h_{31} x_w x_s - h_{32} y_w x_s + h_{11} x_w + h_{12} y_w + h_{13} = h_{33} x_s \tag{2.42}$$

$$-h_{31} x_w y_s - h_{32} y_w y_s + h_{21} x_w + h_{22} y_w + h_{23} = h_{33} y_s. \tag{2.43}$$

An image taken of the calibration plate contains many calibration targets and for each calibration target there exists a pair of equations of the form of Equations 2.42 and 2.43. Furthermore the values of the homography parameters, h_{11} through h_{33}, within these equations remain the same. Therefore a pair of these equations can be defined for each calibration target and these equations can be rearranged into matrix form.

$$\begin{bmatrix} x_{w_1} & y_{w_1} & 1 & 0 & 0 & 0 & -x_{w_1}x_{s_1} & -y_{w_1}x_{s_1} \\ 0 & 0 & 0 & x_{w_1} & y_{w_1} & 1 & -x_{w_1}y_{s_1} & -y_{w_1}y_{s_1} \\ x_{w_2} & y_{w_2} & 1 & 0 & 0 & 0 & -x_{w_2}x_{s_2} & -y_{w_2}x_{s_2} \\ 0 & 0 & 0 & x_{w_2} & y_{w_2} & 1 & -x_{w_2}y_{s_2} & -y_{w_2}y_{s_2} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \end{bmatrix} = h_{33} \times \begin{bmatrix} x_{s_1} \\ y_{s_1} \\ x_{s_2} \\ y_{s_2} \\ \vdots \end{bmatrix} \tag{2.44}$$

This overdetermined system of equations then can be used to solve for the homography matrix of the calibration image using least squares. This is the first step involved in calibration.

In order to avoid the trivial solution of an empty homography matrix , $\boldsymbol{H} = \mathbf{0}$, constraints need to be placed on the elements of the homography matrix. Such a constraint, proposed by Abdel-Aziz [3], suggests setting $h_{33} = 1$. Since the homography matrix is define up to a scale factor this ensures that the trivial solution is avoided without significantly influencing the results. The only risk of applying this constraint is that if the value of h_{33} is close to zero then this constraint will introduce a singularity.

empty correct?

Direct linear transformation normalisation

In practice the use of the DLT method to find estimates to the homographies was found to be sensitive to noise in the images. It was discovered by Hartley [26] that the DLT method can be made much more robust to noise by normalising the sensor and world coordinate points prior to performing DLT.

Specifically each set of coordinates are first translated such that their centroid is at the origin of their coordinate system. Then the coordinates are scaled such that the average distance to the origin is equal to $\sqrt{2}$.

The sets of coordinates are normalised separately with $\boldsymbol{X'_w}$ and $\boldsymbol{X'_s}$ representing the normalised world and sensor coordinate points respectively.

make sure normalised with s or z is used consistently

$$\boldsymbol{X'_w} = \begin{bmatrix} x'_w \\ y'_w \\ 1 \end{bmatrix} = \begin{bmatrix} s_w & 0 & -s_w\bar{x}_w \\ 0 & s_w & -s_w\bar{y}_w \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ 1 \end{bmatrix} = \boldsymbol{T_w} \begin{bmatrix} x_w \\ y_w \\ 1 \end{bmatrix} \tag{2.45}$$

$$\boldsymbol{X'_s} = \begin{bmatrix} x'_s \\ y'_s \\ 1 \end{bmatrix} = \begin{bmatrix} s_s & 0 & -s_s\bar{x}_s \\ 0 & s_s & -s_s\bar{y}_s \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_s \\ y_s \\ 1 \end{bmatrix} = \boldsymbol{T_s} \begin{bmatrix} x_s \\ y_s \\ 1 \end{bmatrix} \tag{2.46}$$

$$\text{where} \quad s_w = \frac{n\sqrt{2}}{\sum_i^n \sqrt{(x_{w_i} - \bar{x}_w)^2 + (y_{w_i} - \bar{y}_w)^2}} \tag{2.47}$$

$$\text{and} \quad s_s = \frac{n\sqrt{2}}{\sum_i^n \sqrt{(x_{s_i} - \bar{x}_s)^2 + (y_{s_i} - \bar{y}_s)^2}} \tag{2.48}$$

Here $(\bar{x}_w, \bar{y}_w)$ and $(\bar{x}_s, \bar{y}_s)$ are the centroids of the respective sets of coordinates and s_w and s_s are the scale factors.

These normalised coordinates are then used in Equation 2.44 thereby reducing the condition number of the matrix and making the equation more robust to noise. Since different coordinate points are used the homography computed, $\boldsymbol{H'}$, is not the homography of the original coordinates, $\boldsymbol{H}$. However the homography calculated can be transformed back in order to obtain the desired homography matrix.

$$\boldsymbol{H} = \boldsymbol{T_s^{-1}} \boldsymbol{H'} \boldsymbol{T_w} \tag{2.49}$$

2.6.5 Absolute conic

As mentioned before two objects that are parallel in Euclidean space appear to intersect each other in projective space. The intersection of these two lines in projective space occurs at a point that lies on the plane at infinity. If a point lies upon the plane at infinity its w is equal to zero in homogeneous coordinates.

The absolute conic lies on the plane at infinity and is defined by the set of points, $\tilde{\boldsymbol{x}}_{ac} = [x, y, z, w]^T$, that satisfies the following.

$$w = 0 \tag{2.50}$$

$$\boldsymbol{x}_{ac}^T \boldsymbol{x}_{ac} = x^2 + y^2 + z^2 = 0 \tag{2.51}$$

Here tilde is used to indicate homogeneous coordinates (projective space) whereas $\boldsymbol{x}_{ac} = [x, y, z]^T$ would be the point in Euclidean space. The absolute conic and its image are illustrated in Figure 2.4

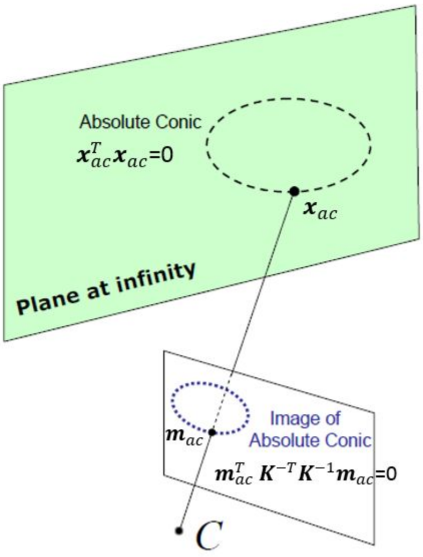

Absolute Conic
$\boldsymbol{x}_{ac}^T \boldsymbol{x}_{ac}$=0
$\boldsymbol{x}_{ac}$
Plane at infinity
$\boldsymbol{m}_{ac}$
Image of
Absolute Conic
$\boldsymbol{m}_{ac}^T \boldsymbol{K}^{-T} \boldsymbol{K}^{-1} \boldsymbol{m}_{ac}$=0
C

Figure 2.4: The absolute conic and the image of the absolute conic [33]

Thus it is this conic of purely imaginary points that lies upon the plane at infinity. The importance of the absolute conic is that it is invariant under any Euclidean transformations. In other words the relative position of the absolute conic to a camera is unaffected by movement of the camera. Consider a point $\boldsymbol{x}_{ac}$ in Euclidean space in the world coordinate system that lies on the absolute conic. It has homogeneous coordinates $\tilde{\boldsymbol{x}}_{ac} = [\boldsymbol{x}_{ac}^T, 0]^T$ in projective space. Applying a rotation and translation to this point to obtain $\tilde{\boldsymbol{x}}'_{ac}$, which is equivalent to moving the camera in the world coordinate system, a corresponding point is obtained in the camera coordinate system.

$$\tilde{\boldsymbol{x}}'_{ac} = \boldsymbol{V}\tilde{\boldsymbol{x}}_{ac} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{T} \\ \boldsymbol{0} & 1 \end{bmatrix} \tilde{\boldsymbol{x}}_{ac} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{ac} \\ y_{ac} \\ z_{ac} \\ 0 \end{bmatrix} \tag{2.52}$$

$$= \begin{bmatrix} \boldsymbol{R}\boldsymbol{x}_{ac} \\ 0 \end{bmatrix} \tag{2.53}$$

It is clear that this point still lies on the plane at infinity since its w is equal to zero. More importantly it can be proven that this point $\boldsymbol{x}'_{ac}$ is on the same absolute conic using the orthogonality of the rotation matrix $\boldsymbol{R}$.

$$\boldsymbol{x}'^T_{ac}\boldsymbol{x}'_{ac} = (\boldsymbol{R}\boldsymbol{x}_{ac})^T (\boldsymbol{R}\boldsymbol{x}_{ac}) = \boldsymbol{x}_{ac}^T \boldsymbol{R}^T \boldsymbol{R}\boldsymbol{x}_{ac} = \boldsymbol{x}_{ac}^T \boldsymbol{R}^{-1}\boldsymbol{R}\boldsymbol{x}_{ac} = \boldsymbol{x}_{ac}^T\boldsymbol{x}_{ac} = 0 \tag{2.54}$$

Thus the absolute conic is invariant to Euclidean transformations. Now consider again a point, $\boldsymbol{x}_{ac}$, lying on the absolute conic. The corresponding point,

$\boldsymbol{m}_{ac}$, in the sensor plane is given according to Equation 2.30

$$\tilde{\boldsymbol{m}}_{ac} = \frac{1}{\alpha}\boldsymbol{K}\boldsymbol{V}\begin{bmatrix}\boldsymbol{x}_{ac}\\ 0\end{bmatrix} = \frac{1}{\alpha}\begin{bmatrix}f_x & f_s & c_x\\ 0 & f_y & c_y\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}\boldsymbol{R} & \boldsymbol{T}\\ \boldsymbol{0} & 1\end{bmatrix}\begin{bmatrix}\boldsymbol{x}_{ac}\\ 0\end{bmatrix} = \frac{1}{\alpha}\boldsymbol{K}\begin{bmatrix}\boldsymbol{R}\boldsymbol{x}_{ac}\\ 0\end{bmatrix} \tag{2.55}$$

$$\therefore \boldsymbol{m}_{ac} = \frac{1}{\alpha}\boldsymbol{K}\boldsymbol{R}\boldsymbol{x}_{ac}. \tag{2.56}$$

Checking whether this point satisfies equation 2.51 results in

$$\boldsymbol{m}_{ac}^T\boldsymbol{K}^{-T}\boldsymbol{K}^{-1}\boldsymbol{m}_{ac} = \frac{1}{\alpha^2}\boldsymbol{x}_{ac}^T\boldsymbol{R}^T\boldsymbol{R}\boldsymbol{x}_{ac} = \frac{1}{\alpha^2}\boldsymbol{x}_{ac}^T\boldsymbol{x}_{ac} = 0 \tag{2.57}$$

Thus the image of the absolute conic is itself an absolute conic. The image of the absolute conic is defined by $\boldsymbol{K}^{-T}\boldsymbol{K}^{-1}$ [19]. Thus since the image of the absolute conic is dependent only on intrinsic camera parameters it can be used to solve for the intrinsic camera parameters.

absolute or imaginary conic?

2.6.6 Constraints on intrinsic parameters

According to Zhang [33] there are two constraints placed upon the intrinsic parameters of the camera. These are important later on in the solving for these intrinsic parameters. The plane of the calibration plate in the camera coordinate system is given by [33]

big t or small t

$$\begin{bmatrix}\boldsymbol{r}_3\\ \boldsymbol{r}_3^T\boldsymbol{T}\end{bmatrix}^T\begin{bmatrix}x_c\\ y_c\\ z_c\\ w\end{bmatrix} = 0. \tag{2.58}$$

Here w is zero for points on the plane at infinity and one for those that are not. This plane intersects the plane at infinity on a line and it happens that $[\boldsymbol{r}_1^T, 0]^T$ and $[\boldsymbol{r}_2^T, 0]^T$ are two points on this line [33]. Thus it is known that any point on this line, $\boldsymbol{x}_c^\infty$, is a linear combination of these two points.

$$\boldsymbol{x}_c^\infty = a\begin{bmatrix}\boldsymbol{r}_1\\ 0\end{bmatrix} + b\begin{bmatrix}\boldsymbol{r}_2\\ 0\end{bmatrix} = \begin{bmatrix}a\boldsymbol{r}_1 + b\boldsymbol{r}_2\\ 0\end{bmatrix} \tag{2.59}$$

Now assume that this point, $\boldsymbol{x}_c^\infty$, lies on the absolute conic. Then this point must satisfy equation 2.51. This would require that $a^2 + b^2 = 0$ which results in the solution $b = \pm ai$, where $i = \sqrt{-1}$. As a result it can be seen that the two points along this line intersect the absolute conic at

$$\boldsymbol{x}_c^\infty = a\begin{bmatrix}\boldsymbol{r}_1 \pm i\boldsymbol{r_2}\\ 0\end{bmatrix}. \tag{2.60}$$

Since these points lie on the absolute conic they are invariant under Euclidean transformations. Their projection, up to a scale factor, in the sensor coordinate system is given by

$$\boldsymbol{x}_s^{\infty} = \boldsymbol{K}\left(\boldsymbol{r}_1 \pm i\boldsymbol{r}_2\right) = \boldsymbol{h}_1 \pm i\boldsymbol{h}_2. \tag{2.61}$$

added i to r in equation make

Here $\boldsymbol{h}_i = [h_{1i}\, h_{2i}\, h_{3i}]^T$ is the i$^{\text{th}}$ column of the homography matrix. Substituting these points into equation 2.51 results in

check that i is correct - italics or not etc

$$\left(\boldsymbol{h}_1 \pm i\boldsymbol{h}_2\right)^T \boldsymbol{K}^{-T}\boldsymbol{K}^{-1}\left(\boldsymbol{h}_1 \pm i\boldsymbol{h}_2\right) = 0. \tag{2.62}$$

Enforcing this equation for both imaginary and real parts results in two constraints on the intrinsic parameters.

$$\boldsymbol{h}_1^T\boldsymbol{K}^{-T}\boldsymbol{K}^{-1}\boldsymbol{h}_2 = 0 \tag{2.63}$$

$$\boldsymbol{h}_1^T\boldsymbol{K}^{-T}\boldsymbol{K}^{-1}\boldsymbol{h}_1 = \boldsymbol{h}_2^T\boldsymbol{K}^{-T}\boldsymbol{K}^{-1}\boldsymbol{h}_2 \tag{2.64}$$

2.6.7 Intrinsic parameters and the absolute conic

Using DLT an homography, $\boldsymbol{H}$, between the calibration plate and the image of the calibration plate can be estimated. This homography can then be used to solve for the image of the absolute conic. Once the image of the absolute conic is known it can be used to solve for the intrinsic camera parameters. The image of the absolute conic can be represented as

$$\boldsymbol{B} = \boldsymbol{K}^{-T}\boldsymbol{K}^{-1} = \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{12} & b_{22} & b_{23} \\ b_{13} & b_{23} & b_{33} \end{bmatrix} \tag{2.65}$$

Since $\boldsymbol{B}$ is symmetric its contents can be represented by a 6D vector $\boldsymbol{b} = [b_{11}\, b_{12}\, b_{22}\, b_{13}\, b_{23}\, b_{33}]^T$. Then using to Zhang's convention [33]

$$\boldsymbol{h}_i^T\boldsymbol{B}\boldsymbol{h}_j = \boldsymbol{q}_{ij}\boldsymbol{b} \tag{2.66}$$

where

$$\boldsymbol{q}_{ij} = [h_{i1}h_{j1}, h_{i1}h_{j2} + h_{i2}h_{j1}, h_{i2}h_{j2}, h_{i3}h_{j1} + h_{i1}h_{j3}, h_{i3}h_{j2} + h_{i2}h_{j3}, h_{i3}h_{j3}]^T. \tag{2.67}$$

Then equations 2.63 and 2.64 can be rewritten as

$$\begin{bmatrix} \boldsymbol{q}_{12}^T \\ (\boldsymbol{q}_{11} - \boldsymbol{q}_{22})^T \end{bmatrix} \boldsymbol{b} = \boldsymbol{0} \tag{2.68}$$

A separate version of this equation exist for each image taken of the calibration plate and these equations can be stacked vertically, in $\boldsymbol{Q}$, to give

$$\boldsymbol{Q}\boldsymbol{b} = \boldsymbol{0}. \tag{2.69}$$

The solution for $\boldsymbol{b}$, up to a scale factor λ, is known to be the eigenvector of $\boldsymbol{Q}^T\boldsymbol{Q}$ associated with the smallest eigenvalue [33]. Once $\boldsymbol{b}$ has been determined it can be used to determine the intrinsic parameters of the matrix $\boldsymbol{K}$. The relation between $\boldsymbol{B}$ and $\boldsymbol{K}$ is $\boldsymbol{B} = \lambda\boldsymbol{K}^{-T}\boldsymbol{K}^{-1}$. The intrinsic parameters are determined as follows.

$$c_y = (b_{12}b_{13} - b_{11}b_{23})/(b_{11}b_{22} - b_{12}^2) \tag{2.70}$$

$$\lambda = b_{33} - (b_{13}^2 + c_y(b_{12}b_{13} - b_{11}b_{23}))/b_{11} \tag{2.71}$$

$$f_x = \sqrt{\frac{\lambda}{b_{11}}} \tag{2.72}$$

$$f_y = \sqrt{\frac{\lambda b_{11}}{b_{11}b_{22} - b_{12}^2}} \tag{2.73}$$

$$f_s = \frac{-b_{12}f_x^2 f_y}{\lambda} \tag{2.74}$$

$$c_x = \frac{f_s c_y}{f_y} - \frac{b_{13}f_x^2}{\lambda} \tag{2.75}$$

Thereafter the extrinsic parameters for each image can be determined separately using the appropriate homography matrix.

$$\boldsymbol{r}_1 = \lambda\boldsymbol{K}^{-1}\boldsymbol{h}_1 \tag{2.76}$$

$$\boldsymbol{r}_2 = \lambda\boldsymbol{K}^{-1}\boldsymbol{h}_2 \tag{2.77}$$

$$\boldsymbol{r}_3 = \boldsymbol{r}_1 \times \boldsymbol{r}_2 \tag{2.78}$$

$$\boldsymbol{T} = \lambda\boldsymbol{K}^{-1}\boldsymbol{h}_3 \tag{2.79}$$

this section is too close to the paper

2.6.8 Distortion in Calibration

Sections 2.6.3 through 2.6.7 outline the framework for determining the closed form solution of the calibration problem when distortion is not considered. The estimates for the extrinsic and intrinsic parameters obtained using this closed form solution need to be improved by accounting for distortion.

As stated in Section 2.5 good calibration results are achieved by only taking radial distortion into account. This is beneficial since it is possible to estimate the radial distortion parameters using a closed form solution in a similar manner to how the intrinsic and extrinsic parameters are estimated.

Taking into account two radial distortion parameters results in distortion being applied to the points on the image plane as

$$\hat{\boldsymbol{X}}_{p_i} = \begin{bmatrix} \hat{x}_{p_i} \\ \hat{y}_{p_i} \end{bmatrix} = \left(1 + \kappa_1 r_i^2 + \kappa_2 r_i^4\right) \begin{bmatrix} x_{p_i} \\ y_{p_i} \end{bmatrix} = \left(1 + \kappa_1 r_i^2 + \kappa_2 r_i^4\right) \boldsymbol{X}_{p_i} \tag{2.80}$$

$$\text{where} \qquad r_i = \sqrt{x_{p_i}^2 + y_{p_i}^2} \tag{2.81}$$

Here $\hat{\boldsymbol{X}}_{p_i} = \begin{bmatrix}\hat{x}_{p_i} & \hat{y}_{p_i}\end{bmatrix}^T$ and $\boldsymbol{X}_{p_i} = \begin{bmatrix}x_{p_i} & y_{p_i}\end{bmatrix}^T$ represent the distorted and undistorted image plane coordinates respectively.

At this point the intrinsic and extrinsic parameters have been determined using the pinhole camera model and the error between the predicted calibration plate targets and the actual location of these in the images is attributed to radial distortion [12]. This error is represented as the observed distortion vector $\boldsymbol{d}_{1_i}$.

$$\boldsymbol{d}_{1_i} = \boldsymbol{X}_{s_i}^a - \boldsymbol{X}_{s_i}^c \tag{2.82}$$

Here $\boldsymbol{X}_{s_i}^a$ is the actual coordinates observed and $\boldsymbol{X}_{s_i}^c$ is the predicted coordinates that are calculated according to the camera model. These coordinates are in the sensor coordinate system.

check bolding of vectors is correct

According to Burger [12] an undistorted point in the sensor coordinate system, $\boldsymbol{X}_{s_i}^c$ is distorted to point $\hat{\boldsymbol{X}}_{s_i}^c$ as

$$\begin{aligned}
\hat{\boldsymbol{X}}_{s_i}^c &= \boldsymbol{U}_c + (\boldsymbol{X}_{s_i}^c - \boldsymbol{U}_c)(1 + \kappa_1 r_i^2 + \kappa_2 r_i^4) && (2.83)\\
&= \boldsymbol{U}_c + \boldsymbol{X}_{s_i}^c - \boldsymbol{U}_c + (\boldsymbol{X}_{s_i}^c - \boldsymbol{U}_c)(\kappa_1 r_i^2 + \kappa_2 r_i^4) && (2.84)\\
&= \boldsymbol{X}_{s_i}^c + (\boldsymbol{X}_{s_i}^c - \boldsymbol{U}_c)(\kappa_1 r_i^2 + \kappa_2 r_i^4) && (2.85)\\
&= \boldsymbol{X}_{s_i} + \boldsymbol{d}_{2_i} && (2.86)
\end{aligned}$$

Here $\boldsymbol{U}_c = \begin{bmatrix}u_c & v_c\end{bmatrix}^T$ is the projection centre or principal point (the intersection of the optical axis with the image plane) in the sensor coordinate system and is determined by applying Equation 2.26 to the principal point, $(0, 0)$, given in terms of the image plane coordinate system. The term $\boldsymbol{d}_{2_i}$ is referred to as the model distortion vector. Note that r_i used here is calculated according to Equation 2.81 using image plane coordinates.

The distortion parameters can be estimated by minimizing the difference between the model and observed distortion vectors. Thus the least squares solution to the overdetermined system of equations of the form $\boldsymbol{d}_{2_i} = \boldsymbol{d}_{1_i}$ is what is needed. For each point $\boldsymbol{X}_{s_i}^c$ considered there are two equations of the form

$$\begin{aligned}
(x_{s_i}^c - u_c) \times (\kappa_1 r_i^2 + \kappa_2 r_i^4) &= (x_{s_i}^a - x_{s_i}^c) && (2.87)\\
(y_{s_i}^c - v_c) \times (\kappa_1 r_i^2 + \kappa_2 r_i^4) &= (y_{s_i}^a - y_{s_i}^c). && (2.88)
\end{aligned}$$

Two equations of this form are created for each target of the calibration plate and these equations are then stacked into a system of equations which is used to solved for the least squares solution to the distortion parameters.

$$\begin{bmatrix} (x^c_{s_1} - u_c) \times r_1^2 & (x^c_{s_1} - u_c) \times r_1^4 \\ (y^c_{s_1} - v_c) \times r_1^2 & (y^c_{s_1} - v_c) \times r_1^4 \\ (x^c_{s_2} - u_c) \times r_2^2 & (x^c_{s_2} - u_c) \times r_2^4 \\ (y^c_{s_2} - v_c) \times r_2^2 & (y^c_{s_2} - v_c) \times r_2^4 \\ \vdots & \vdots \end{bmatrix} \begin{bmatrix} \kappa_1 \\ \kappa_2 \end{bmatrix} = \begin{bmatrix} x^a_{s_1} - x^c_{s_1} \\ y^a_{s_1} - y^c_{s_1} \\ x^a_{s_2} - x^c_{s_2} \\ y^a_{s_2} - y^c_{s_2} \\ \vdots \end{bmatrix} \tag{2.89}$$

2.6.9 Non-linear optimisation

At this point good approximations have been calculated for all the calibration parameters using closed form solution methods. However, since the correlation process relies upon the calibration process to link optical flow in images to displacement in the 3D world, these calibration parameters need to be refined so that they don't negatively affect the correlation results.

First Equation 2.30 must be modified to incorporate radial distortion. Radial distortion is applied to the image plane coordinates prior to converting them to sensor coordinates. As such the conversion from world coordinates, $\boldsymbol{X}_w$, to distorted sensor coordinates, $\hat{\boldsymbol{X}}_s$, can be broken down into three functions as

$$\hat{\boldsymbol{X}}_s = f_{ws}(\boldsymbol{X}_w, \boldsymbol{R}, \boldsymbol{T}, \gamma, \kappa_1, \kappa_2, S_x, S_y, \phi, \hat{c}_x, \hat{c}_y) \tag{2.90}$$

$$= f_{is}(f_{dist}(f_{wi}(\boldsymbol{X}_w, \boldsymbol{R}, \boldsymbol{T}, \gamma), \kappa_1, \kappa_2), S_x, S_y, \phi, \hat{c}_x, \hat{c}_y) \tag{2.91}$$

where f_{ws} represents the function that converts from world to sensor coordinates, f_{is} converts image plane coordinates to sensor coordinates, f_{dist} distorts the image plane coordinates and f_{wi} converts from world to image plane coordinates.

These functions are expanded as

$$f_{wi}(\boldsymbol{X}_w, \boldsymbol{R}, \boldsymbol{T}, \gamma) = \boldsymbol{X}_p = \begin{bmatrix} \gamma & 0 & 0 & 0 \\ 0 & \gamma & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{R} & \boldsymbol{T} \\ \boldsymbol{0} & 1 \end{bmatrix} \boldsymbol{X}_w \tag{2.92}$$

$$f_{dist}(\boldsymbol{X}_p, \kappa_1, \kappa_2) = \hat{\boldsymbol{X}}_p = \left(1 + \kappa_1(\boldsymbol{X}_p^T \boldsymbol{X}_p) + \kappa_2(\boldsymbol{X}_p^T \boldsymbol{X}_p)^2\right) \boldsymbol{X}_p \tag{2.93}$$

$$f_{is}(\hat{\boldsymbol{X}}_p, S_x, S_y, \phi, \hat{c}_x, \hat{c}_y) = \hat{\boldsymbol{X}}_s = \begin{bmatrix} S_x & -S_x \cot\phi & -S_x\left(\hat{c}_x - \hat{c}_y \cot\phi\right) \\ 0 & \frac{S_y}{\sin\phi} & -\frac{S_y \hat{c}_y}{\sin\phi} \\ 0 & 0 & 1 \end{bmatrix} \hat{\boldsymbol{X}}_p. \tag{2.94}$$

This function, f_{ws}, is applied to the known calibration targets in the world coordinate system, $\boldsymbol{X}_w^a$, in order to predict where they will appear in the image, $\boldsymbol{X}_s^c$. However, since estimates of the intrinsic, extrinsic and distortion parameters are used, these calculated locations will differ from the actual locations of the calibration targets, $\boldsymbol{X}_s^a$. The absolute value of these differences are summed together to obtain the projection error, P_{error}.

$$P_{error} = \sum_i^n \left|\boldsymbol{X}_{s_i}^a - \boldsymbol{X}_{s_i}^c\right| = \sum_i^n \left|\boldsymbol{X}_{s_i}^a - f_{ws}(\boldsymbol{X}_{w_i}^a, \boldsymbol{R}, \boldsymbol{T}, \gamma, \kappa_1, \kappa_2, S_x, S_y, \phi, \hat{c}_x, \hat{c}_y)\right| \tag{2.95}$$

Consequently these estimated parameters need to be optimised using an iterative non-linear optimisation technique to complete the calibration process. Matlab's 'lsqnonlin' optimisation algorithm was chosen for this project. The aim of using this optimisation algorithm is to minimise the projection error by iteratively making improvements to the estimated parameters.

2.7 Correspondence problem

The purpose of DIC is to solve the correspondence problem between two images. The correspondence problem refers to solving for the apparent motion that an object, within a set of images, experiences from one image to the next. For example consider a traffic camera that records images of a car at an intersection. If someone were to review these images the apparent motion of the car between images would be obvious.

This is because humans are incredibly adapt at solving the correspondence problem through pattern recognition. Computers can also be used to solve the correspondence problem, and to much greater accuracy than a human, however they need the correspondence problem to be broken down into a well defined mathematical problem. The way in which the correspondence problem is broken down into a mathematical problem determines what type of DIC algorithm is used to solve it.

This project is focused on using DIC for material testing where the deformation experienced by a loaded specimen needs to be measured. As such it is not the overall motion of the specimen that is sought for but the motion that portions of the specimen experience as a result of the stresses within the specimen. Therefore in this project the correspondence problem is broken down as tracking the pattern contained within a cluster of neighbouring pixels from one image to the next.

This cluster of neighbouring pixels is termed a subset and a subset spans a small portion of the specimen being imaged. Thus, by tracking the motion of many subsets that together span the whole surface of the specimen, the deformation of the specimen can be measured. The DIC algorithms that are used to solve this type of correspondence problem are termed subset-based DIC algorithms.

2.7.1 Speckle pattern

Tracking of subsets in this way places requirements on the patterns contained within the subsets. To illustrate this consider a checker board pattern which is a repeated pattern of alternating black and white squares of constant size. An image is taken of this pattern in the reference position and another is taken after the pattern has been displaced to the right by a distance equal to two and a half times the length of one of the sides of the squares.

Treating the whole first image as a single subset the displacement between the two images can be calculated. However this calculated displacement will be half the length of one of the sides of the squares. This is because if the first image is displaced by this amount it will be identical to the second image even though the true displacement between the images is much greater.

Thus it is clear that the pattern to be tracked in the images can have a significant impact on the performance of the correlation process. According to Sutton [27] for the best results the pattern contained within the images should be isotropic and non-periodic. This is achieved by applying a random speckle pattern to the surface of the specimen to be tested since these provide very unique patterns which are isotropic in nature. Additionally their high information density enables smaller subsets to be used which allows the deformation of the specimen to be measured with more detail.

corr problem subsets speckle patterns

2.8 Correlation

Correlation is the process of solving the correspondence problem between two images. According to how the correspondence problem was broken down in Section 2.7 correlation determines how the pattern within a subset of one image, the reference image, needs to be displaced and deformed such that it matches the corresponding pattern in another image, the deformed image.

However the correspondence problem first needs to be mathematically defined before it can be solved. There are four elements that together mathematically define the correspondence problem. These are the warp function, correlation criteria, interpolation method and optimisation algorithm. These elements are discussed in the sections that follow.

2.8.1 Warp function

When applying DIC to material science the deformation behaviour of materials under load must be taken into account. This is because the deformation of the material causes the speckle pattern on its surface to deform in the same way. This is an issue since the pattern contained within a subset of the reference image will be deformed in the second image which can cause difficulties in matching the two subsets.

This is solved by allowing the subsets to deform in a similar way to that of the material by defining the allowable deformations in a warp function. The purpose of the warp function is to transform the pixel coordinates of a subset in one image so that the resulting pattern is closer to the pattern contained in another image. The most popular warp function in material science is the first order warp function which accounts for affine transformations. It is popular because its parameters are closely related to the strains used to characterise material deformation. Thus by determining the warp function parameters for a specific subset; the strains experienced by that subset are also determined. This warp function is given as

$$W(\boldsymbol{\zeta}, \boldsymbol{p}) = \begin{bmatrix} \Delta x_{warp} \\ \Delta y_{warp} \end{bmatrix} = \begin{bmatrix} 1 + \frac{\partial u}{\partial x} & \frac{\partial u}{\partial y} & u \\ \frac{\partial v}{\partial x} & 1 + \frac{\partial v}{\partial y} & v \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ 1 \end{bmatrix}. \tag{2.96}$$

Here u and v represent the displacements in the x and y directions respectively, $\frac{\partial u}{\partial x}$ and $\frac{\partial v}{\partial y}$ represent the elongation in the x and y directions respectively whereas $\frac{\partial v}{\partial x}$ and $\frac{\partial u}{\partial y}$ represent the shear deformation of the subset. All of these variables are stored within the vector $\boldsymbol{p}$ and are referred to as the warp function parameters (WFPs). The WFPs define how the subset is warped and are stored as

$$\boldsymbol{p} = \begin{bmatrix} u & \frac{\partial u}{\partial x} & \frac{\partial u}{\partial y} & v & \frac{\partial v}{\partial x} & \frac{\partial v}{\partial y} \end{bmatrix} = \begin{bmatrix} p_1 & p_2 & p_3 & p_4 & p_5 & p_6 \end{bmatrix} \tag{2.97}$$

Additionally Δx and Δy are the distances from the centre of the undeformed subset to the pixel under consideration and are contained within the variable $\boldsymbol{\zeta} = \begin{bmatrix} \Delta x & \Delta y & 1 \end{bmatrix}^T$. The outputs Δx_{warp} and Δy_{warp} are the modified distances from the centre of the subset to the pixel under consideration. Thus it is the pixel locations within the subset that are modified in order to induce warp and not the subset itself which is warped by this function. In other words the subset's shape remains the same but the pattern contained within the subset warps as a results of sampling the light intensity values of the image at warped locations. Thus the warp function applied to a subset is mathematically expressed as

$$G_{warped} = G(\boldsymbol{x_0} + W(\boldsymbol{\zeta}, \boldsymbol{p})) \tag{2.98}$$

where $\boldsymbol{x_0} = \begin{bmatrix} x_0 & y_0 \end{bmatrix}^T$ are the x and y coordinates, in units of pixels, of the centre of the subset.

2.8.2 Correlation criteria

A correlation criteria is a mathematical way of quantifying how well a subset in one image matches a subset in another image. It is used to calculate the correlation coefficient which indicates whether the WFPs, that have been solved for, are sufficiently accurate to explain how a subset in the reference image is related to the deformed image. Furthermore the correlation criteria is usually used as the basis for the optimisation algorithm that performs the correlation process. As such it is important that the correlation criteria reliably indicates the fit between the two subsets.

This reliability refers to dealing with the changes in lighting between images. As images are taken, as the specimen is deformed, the light intensity of the images can change as a result of changes in lighting, changes in reflectivity as the material strains and so on. Although this does not cause the patterns in the subsets to change it does cause the grayscale values of individual pixels, that make up these patterns, to change which can negatively affect the correlation process. The changes in lighting that takes place between images have been categorised into two types; offset and scaling.

The two most commonly used correlation criteria are the Cross-Correlation and Sum of Squared difference criteria. Each of these has four versions for dealing with various combinations of light intensity changes.

Cross-Correlation

The cross-correlation criteria obtains a coefficient by multiplying each element of the reference subset with its corresponding element in the investigated subset, of the deformed image, and then summing all of these values together as shown below. Thus a higher value means better correlation.

$$C_{CC} = \sum_{i}^{n} F_i G_i \tag{2.99}$$

Here n is the number of pixels contained within a subset. This correlation criteria can be modified to be capable of dealing with offsets in light intensity by using zero-mean normalisation. Zero-mean normalisation determines the mean intensity value of a subset and then subtracts that mean intensity from every pixel in that subset. This is done for both subsets to obtain the Zero-Mean Cross-Correlation as shown below.

$$C_{ZCC} = \sum_{i}^{n} \left(F_i - \bar{F}\right)\left(G_i - \bar{G}\right) \tag{2.100}$$

$$\text{where} \quad \bar{F} = \frac{\sum_i^n F_i}{n} \quad \text{and} \quad \bar{G} = \frac{\sum_i^n G_i}{n} \tag{2.101}$$

The correlation criteria can also be modified to deal with scaling of light intensities by dividing it by the root-sum-square of the subset's intensities. This is referred to as Normalised Cross-Correlation.

$$C_{NCC} = \frac{\sum_i^n F_i G_i}{\sqrt{\sum_i^n F_i^2 \sum_i^n G_i^2}} \tag{2.102}$$

Both of these methods can be combined together to obtain Zero-Mean Normalised Cross-Correlation which is insensitive to both scaling and offset.

$$C_{ZNCC} = \frac{\sum_i^n (F_i - \bar{F})(G_i - \bar{G})}{\sqrt{\sum_i^n (F_i - \bar{F})^2 \sum_i^n (G_i - \bar{G})^2}} \tag{2.103}$$

Sum of Squared Difference

As the name suggests this correlation criteria is based on summing the square of the differences between the light intensities of the reference and the investigated subsets.

$$C_{SSD} = \sum_i^n (F_i - G_i)^2 \tag{2.104}$$

Since this correlation criteria measures differences; a smaller value means better correlation between the subsets. This correlation criteria can be modified in the same way to become insensitive to both offset (Zero-Mean Sum of Squared Difference) and scaling (Normalised Sum of Squared Difference).

$$C_{ZSSD} = \sum_i^n \left[\left(F_i - \bar{F}\right) - \left(G_i - \bar{G}\right)\right]^2 \tag{2.105}$$

$$C_{NSSD} = \sum_i^n \left[\frac{F_i}{\sqrt{\sum_i^n F_i^2}} - \frac{G_i}{\sqrt{\sum_i^n G_i^2}}\right]^2 \tag{2.106}$$

These can then be combined to obtain the Zero-Mean Normalised Sum of Squared Difference correlation criteria which is insensitive to both scaling and offset in light intensities.

$$C_{ZNSSD} = \sum_i^n \left[\frac{F_i - \bar{F}}{\sqrt{\sum_i^n (F_i - \bar{F})^2}} - \frac{G_i - \bar{G}}{\sqrt{\sum_i^n (G_i - \bar{G})^2}}\right]^2 \tag{2.107}$$

Relation between ZNCC and ZNSSD

It can be shown that the Zero-Mean Normalised Cross-Correlation and Zero-Mean Normalised Sum of Squared Difference correlation criteria are related.

$$C_{ZNSSD} = \sum_i^n \left[\frac{F_i - \bar{F}}{\sqrt{\sum_i^n (F_i - \bar{F})^2}} - \frac{G_i - \bar{G}}{\sqrt{\sum_i^n (G_i - \bar{G})^2}}\right]^2 \tag{2.108}$$

$$= \sum_{i}^{n} \left(\frac{(F_i - \bar{F})^2}{\sum_i^n (F_i - \bar{F})^2} - 2\frac{(F_i - \bar{F})(G_i - \bar{G})}{\sqrt{\sum_i^n (F_i - \bar{F})^2}\sqrt{\sum_i^n (G_i - \bar{G})^2}} + \frac{(G_i - \bar{G})^2}{\sum_i^n (G_i - \bar{G})^2} \right) \tag{2.109}$$

$$= \frac{\sum_i^n (F_i - \bar{F})^2}{\sum_i^n (F_i - \bar{F})^2} + \frac{\sum_i^n (G_i - \bar{G})^2}{\sum_i^n (G_i - \bar{G})^2} - 2\frac{\sum_i^n (F_i - \bar{F})(G_i - \bar{G})}{\sqrt{\sum_i^n (F_i - \bar{F})^2}\sqrt{\sum_i^n (G_i - \bar{G})^2}} \tag{2.110}$$

$$= 2\left(1 - C_{ZNCC}\right) \tag{2.111}$$

This is significant since the ZNSSD criteria is easier to calculate while the ZNCC criteria value is easier to interpret. Thus it is common practice to calculate the ZNSSD criteria coefficient and convert it to the ZNCC value for easier interpretation.

slightly different from Czncc, make sure equal

2.8.3 Interpolation

It is clear that warping a subset according to equation 2.98 requires that the light intensity values between pixel locations be determined. This is an issue because images store this information in a discrete manner. This is solved by using interpolation to determine the light intensity values between pixels based on the light intensity values of neighbouring pixels. There are multiple interpolation algorithms that can be employed for this purpose each offering varying degrees of accuracy and computational efficiency.

Computational efficiency is important since image interpolation is typically the most time consuming part of the correlation process for iterative sub-pixel DIC algorithms [24]. Additionally the interpolation algorithm needs to be accurate enough to give reliable estimates such that it does not itself introduce bias into the warped subset it calculates. With this taken into account the bicubic interpolation algorithm was chosen since it offers a good balance between computational efficiency and accuracy.

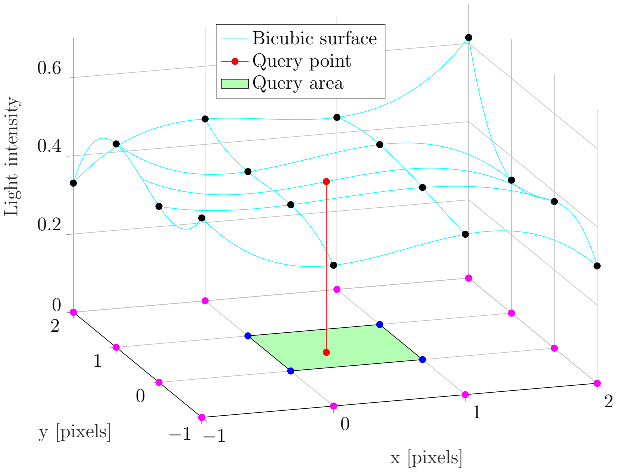


Figure 2.5: The bicubic interpolation surface

add x_{fq} to plot

Bicubic interpolation attempts to fit a fourth order, two-dimensional polynomial surface to the known light intensity values at integer pixel locations as shown in Figure 2.5. Interpolation is performed by substituting the fractional part of the query point into the equation for this surface. The fractional parts of the query points are used because of the way in which the coordinate system of the interpolation surface is defined with the neighbouring pixels of the query point falling on coordinates $(0, 0)$, $(0, 1)$, $(1, 0)$ and $(1, 1)$.

To relate the query point to its fractional part the concept of the floor function needs to be introduced. The floor function takes in a number, z, and returns the largest integer less than or equal to it. The function is indicated as

$$floor(z) = \lfloor z \rfloor \tag{2.112}$$

Therefore the query point, (x_q, y_q), is related to the fractional part of the query point, (x_{fq}, y_{fq}), as

$$x_{fq} = x_q - \lfloor x_q \rfloor \quad \text{and} \quad y_{fq} = y_q - \lfloor y_q \rfloor \tag{2.113}$$

Thus interpolation is performed as

$$F(x, y) = \sum_{i=1}^{4} \sum_{j=1}^{4} a_{ij} \times x_{fq}^{i-1} y_{fq}^{j-1} \tag{2.114}$$

where a_{ij} are the sixteen coefficients of the interpolation equation which define the interpolation surface.

These coefficients are determined based on the light intensities and light intensity gradients at the four integer pixel locations that enclose the query area, hereafter referred to as the query pixels. These locations are depicted as blue in Figure 2.5. The gradients, which are partial derivatives in the x, y and x-y directions, are determined using the central difference numerical differentiation formula as [16]

$$\frac{\partial F(x_k, y_v)}{\partial x} = \frac{F(x_{k+1}, y_v) - F(x_{k-1}, y_v)}{2} \tag{2.115}$$

$$\frac{\partial F(x_k, y_v)}{\partial y} = \frac{F(x_k, y_{v+1}) - F(x_k, y_{v-1})}{2} \tag{2.116}$$

$$\frac{\partial^2 F(x_k, y_v)}{\partial x \partial y} = \frac{[F(x_{k+1}, y_{v+1}) - F(x_{k-1}, y_{v+1})] - [F(x_{k+1}, y_{v-1}) - F(x_{k-1}, y_{v-1})]}{4} \tag{2.117}$$

where k and v are indices which indicate which pixels are used. Applying Equations 2.115, 2.116 and 2.117 to the query pixels, such that $k = 1, 2$ and $v = 1, 2$, gives twelve gradients. To relate these twelve gradients to the coefficients the Equation 2.114 is differentiated in the x, y and x-y directions. Then the appropriate query pixels' locations are substituted into the right-hand side (RHS) and the calculated gradients into the left-hand side (LHS) of these equations. The remaining four relations needed are obtained using Equation 2.114 by simply substituting the query pixels' locations into the RHS and the light intensity of the query pixels into the LHS of the equation. This results in sixteen equations with sixteen unknowns which can be solved simultaneously to determine the coefficients.

However by performing this whole process symbolically, from differentiation to solving the simultaneous equations, an equation can be determined for each coefficients as a function of the known integer pixel light intensities. These equations are (Note: will put this in the appendices)

$$a_{11} = F(x_2, y_2) \tag{2.118}$$

$$a_{12} = -0.5 \times F(x_2, y_1) + 0.5 \times F(x_2, y_3) \tag{2.119}$$

$$a_{13} = F(x_2, y_1) - 2.5 \times F(x_2, y_2) + 2 \times F(x_2, y_3) - 0.5 \times F(x_2, y_4) \tag{2.120}$$

$$a_{14} = -0.5 \times F(x_2, y_1) + 1.5 \times F(x_2, y_2) - 1.5 \times F(x_2, y_3) + 0.5 \times F(x_2, y_4) \tag{2.121}$$

$$a_{21} = -0.5 \times F(x_1, y_2) + 0.5 \times F(x_3, y_2) \tag{2.122}$$

$$a_{22} = 0.25 \times F(x_1, y_1) - 0.25 \times F(x_1, y_3) - 0.25 \times F(x_3, y_1) + 025 \times F(x_3, y_3) \tag{2.123}$$

$$\begin{aligned} a_{23} = {} & -0.5 \times F(x_1, y_1) + 1.25 \times F(x_1, y_2) - F(x_1, y_3) + 0.25 \times F(x_1, y_4) \\ & + 0.5 \times F(x_3, y_1) - 1.25 \times F(x_3, y_2) + F(x_3, y_3) - 0.25 \times F(x_3, y_4) \end{aligned} \tag{2.124}$$

$$a_{24} = 0.25 \times F(x_1, y_1) - 0.75 \times F(x_1, y_2) + 0.75 \times F(x_1, y_3) - 0.25 \times F(x_1, y_4)$$

$$- 0.25 \times F(x_3, y_1) + 0.75 \times F(x_3, y_2) - 0.75 \times F(x_3, y_3) + 0.25 \times F(x_3, y_4) \tag{2.125}$$

$$a_{31} = F(x_1, y_2) - 2.5 \times F(x_2, y_2) + 2 \times F(x_3, y_2) - 0.5 \times F(x_4, y_2) \tag{2.126}$$

$$\begin{aligned} a_{32} = &-0.5 \times F(x_1, y_1) + 0.5 \times F(x_1, y_3) + 1.25 \times F(x_2, y_1) - 1.25 \times F(x_2, y_3) \\ &- F(x_3, y_1) + F(x_3, y_3) + 0.25 \times F(x_4, y_1) - 0.25 \times F(x_4, y_3) \end{aligned} \tag{2.127}$$

$$\begin{aligned} a_{33} = &F(x_1, y_1) - 2.5 \times F(x_1, y_2) + 2 \times F(x_1, y_3) - 0.5 \times F(x_1, y_4) \\ &- 2.5 \times F(x_2, y_1) + 6.25 \times F(x_2, y_2) - 5 \times F(x_2, y_3) + 1.25 \times F(x_2, y_4) \\ &+ 2 \times F(x_3, y_1) - 5 \times F(x_3, y_2) + 4 \times F(x_3, y_3) - F(x_3, y_4) - 0.5 \times F(x_4, y_1) \\ &+ 1.25 \times F(x_4, y_2) - F(x_4, y_3) + 0.25 \times F(x_4, y_4) \end{aligned} \tag{2.128}$$

$$\begin{aligned} a_{34} = &-0.5 \times F(x_1, y_1) + 1.5 \times F(x_1, y_2) - 1.5 \times F(x_1, y_3) + 0.5 \times F(x_1, y_4) \\ &+ 1.25 \times F(x_2, y_1) - 3.75 \times F(x_2, y_2) + 3.75 \times F(x_2, y_3) - 1.25 \times F(x_2, y_4) \\ &- F(x_3, y_1) + 3 \times F(x_3, y_2) - 3 \times F(x_3, y_3) + F(x_3, y_4) + 0.25 \times F(x_4, y_1) \\ &- 0.75 \times F(x_4, y_2) + 0.75 \times F(x_4, y_3) - 0.25 \times F(x_4, y_4) \end{aligned} \tag{2.129}$$

$$a_{41} = -0.5 \times F(x_1, y_2) + 1.5 \times F(x_2, y_2) - 1.5 \times F(x_3, y_2) + 0.5 \times F(x_4, y_2) \tag{2.130}$$

$$\begin{aligned} a_{42} = &0.25 \times F(x_1, y_1) - 0.25 \times F(x_1, y_3) - 0.75 \times F(x_2, y_1) + 0.75 \times F(x_2, y_3) \\ &+ 0.75 \times F(x_3, y_1) - 0.75 \times F(x_3, y_3) - 0.25 \times F(x_4, y_1) + 0.25 \times F(x_4, y_3) \end{aligned} \tag{2.131}$$

$$\begin{aligned} a_{43} = &-0.5 \times F(x_1, y_1) + 1.25 \times F(x_1, y_2) - F(x_1, y_3) + 0.25 \times F(x_1, y_4) \\ &+ 1.5 \times F(x_2, y_1) - 3.75 \times F(x_2, y_2) + 3 \times F(x_2, y_3) - 0.75 \times F(x_2, y_4) \\ &- 1.5 \times F(x_3, y_1) + 3.75 \times F(x_3, y_2) - 3 \times F(x_3, y_3) + 0.75 \times F(x_3, y_4) \\ &+ 0.5 \times F(x_4, y_1) - 1.25 \times F(x_4, y_2) + F(x_4, y_3) - 0.25 \times F(x_4, y_4) \end{aligned} \tag{2.132}$$

$$\begin{aligned} a_{44} = &0.25 \times F(x_1, y_1) - 0.75 \times F(x_1, y_2) + 0.75 \times F(x_1, y_3) - 0.25 \times F(x_1, y_4) \\ &- 0.75 \times F(x_2, y_1) + 2.25 \times F(x_2, y_2) - 2.25 \times F(x_2, y_3) + 0.75 \times F(x_2, y_4) \\ &+ 0.75 \times F(x_3, y_1) - 2.25 \times F(x_3, y_2) + 2.25 \times F(x_3, y_3) - 0.75 \times F(x_3, y_4) \\ &- 0.25 \times F(x_4, y_1) + 0.75 \times F(x_4, y_2) - 0.75 \times F(x_4, y_3) + 0.25 \times F(x_4, y_4) \end{aligned} \tag{2.133}$$

2.8.4 Optimisation and the correlation criterion

The correspondence problem is solved by finding the optimal WFPs which when applied to the deformed subset G minimise the correlation criterion. The optimisation algorithms considered in this project are gradient-descent based algorithms which iteratively improve upon an initial guess of the WFPs using the correlation criterion as the cost function. As such the correlation criterion needs to be a function of the WFPs. This is simply a matter of replacing G_i in the correlation criterion Equations 2.104, 2.105, 2.106 and 2.107 with the warped light intensity values given in Equation 2.98 as

$$G_i = G_{warped} = G(\boldsymbol{x}_0 + W(\boldsymbol{\zeta}_i, \boldsymbol{p})). \tag{2.134}$$

Here $G(\boldsymbol{x}_0 + W(\boldsymbol{\zeta}_i, \boldsymbol{p}))$ indicates interpolation to find the light intensity value at the warped pixel location $\boldsymbol{x}_0 + W(\boldsymbol{\zeta}_i, \boldsymbol{p})$. Similarly the variable F_i actually represents

$$F_i = F(\boldsymbol{x}_0 + \boldsymbol{\zeta}_i). \tag{2.135}$$

Here no interpolation is involved because the pixel locations are not warped. Although all the variables G_i and F_i in these correlation criterion equations should be replaced by their expanded forms given in Equations 2.134 and 2.135, the variables G_i and F_i are often kept in order to make the correlation criterion equations easier to read. Thus it should be kept in mind what these variables represent.

Chapter 3

Optimisation

This chapter derives the optimisation algorithm which is used to solve the correspondence problem. First the Newton-Raphson algorithm is presented followed by the Guass-Newton algorithm which is a simplification of the Newton-Raphson algorithm.

In this chapter the optimisation algorithm which solves the correspondence problem is derived. First the Newton-Raphson algorithm is presented which

In this chapter the optimisation algorithms that solve the correspondence problem are presented. First the Newton-Raphson method is presented. Thereafter the Gauss-Newton algorithm, which is a simplification of the Newton-Raphson algorithm, is presented. This algorithm is focused on in this project. Finally a subset splitting method is presented which allows the Gauss-Newton algorithm to account for a displacement discontinuity within a subset.

The purpose of presenting the Newton-Raphson algorithm is to show the mathematical reasoning behind the Gauss-Newton algorithm and to illustrate how the two algorithms are related.

3.1 Newton-Raphson

The Newton-Raphson method is one of the most popular methods for finding the roots of a function. Its uses the first two terms of the Taylor series approximation of a function to iteratively improve upon an initial estimate, x_i, to the root of a function. The Taylor series expansion of a function is given as

$$f(x_{i+1}) = f(x_i) + f'(x_i) \times (x_{i+1} - x_i) + \frac{f''(x_i)}{2!} \times (x_{i+1} - x_i)^2 + ... \tag{3.1}$$

The Newton-Raphson method is derived by considering only the first two terms of the Taylor series and setting $f(x_{i+1}) = 0$.

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)} \tag{3.2}$$

Thus this method uses the tangent line of the function, at the current estimate x_i, to find an improved estimate x_{i+1} to the root of the function. The improved estimate is the intersection point of the tangent line with the x-axis. However for the purpose of solving the correspondence problem this method needs to be modified to find the minimum of a function and not the root since it is unlikely that a root exists. A root of a correlation criterion does not exist since that would indicate perfect correlation which is not possible due to errors introduced in the correlation process as a result of noise in the image and interpolation used to sample light intensity values between pixel locations.

Since it is known that a minimum or maximum of a function exists where the derivative of the function is equal to zero the Newton-Raphson method is modified by replacing the function, $f(x)$, with its derivative, $f'(x)$, in Equation 3.1.

$$f'(x_{i+1}) = f'(x_i) + f''(x_i) \times (x_{i+1} - x_i) + \frac{f'''(x_i)}{2!} \times (x_{i+1} - x_i)^2 + ... \quad (3.3)$$

Then by considering only the first two terms of the Taylor series expansion and setting $f'(x_{i+1}) = 0$ results in an equation to find the minimum of the function $f(x)$.

$$x_{i+1} = x_i - \frac{f'(x_i)}{f''(x_i)} \quad (3.4)$$

In order to solve the correspondence problem the correlation criterion equation, $C(\boldsymbol{p})$, is to be minimised in terms of the WFPs, $\boldsymbol{p}$. This can be done using Equation 3.4 where x_i is replaced by the current estimate of the WFPs, $\boldsymbol{p}_i$, and the function, $f(x)$, is replaced by the correlation criterion equation, $C(\boldsymbol{p})$. Thus an improved estimate of the WFPs, $\boldsymbol{p}_{i+1}$, is found as

$$\boldsymbol{p}_{i+1} = \boldsymbol{p}_i - \frac{C'(\boldsymbol{p}_i)}{C''(\boldsymbol{p}_i)} = \boldsymbol{p}_i - \frac{\frac{\partial C(\boldsymbol{p}_i)}{\partial \boldsymbol{p}}}{\frac{\partial^2 C(\boldsymbol{p}_i)}{\partial \boldsymbol{p} \partial \boldsymbol{p}}}. \quad (3.5)$$

Here $\frac{\partial C(\boldsymbol{p}_i)}{\partial \boldsymbol{p}}$ and $\frac{\partial^2 C(\boldsymbol{p}_i)}{\partial \boldsymbol{p} \partial \boldsymbol{p}}$ are the Jacobian, $\boldsymbol{J}$, and Hessian, $\boldsymbol{H}$, of the correlation criterion, in terms of the WFPs, at the current estimate of the WFPs. These derivatives of the correlation criterion are of the form of a vector and matrix because the correlation criterion is a mutlivariable function. For the general case of six WFPs, for a first order warp function, the Jacobian is of the form

$$\boldsymbol{J} = \frac{\partial C(\boldsymbol{p})}{\partial \boldsymbol{p}} = \begin{bmatrix} \frac{\partial C(\boldsymbol{p})}{\partial p_1} & \frac{\partial C(\boldsymbol{p})}{\partial p_2} & \frac{\partial C(\boldsymbol{p})}{\partial p_3} & \frac{\partial C(\boldsymbol{p})}{\partial p_4} & \frac{\partial C(\boldsymbol{p})}{\partial p_5} & \frac{\partial C(\boldsymbol{p})}{\partial p_6} \end{bmatrix}^T. \quad (3.6)$$

Similarly the Hessian is of the form

make sure jacobian vector is this orientation

$$\boldsymbol{H} = \frac{\partial^2 C(\boldsymbol{p})}{\partial \boldsymbol{p} \partial \boldsymbol{p}} = \begin{bmatrix} \frac{\partial^2 C(\boldsymbol{p})}{\partial p_1^2} & \frac{\partial^2 C(\boldsymbol{p})}{\partial p_1 \partial p_2} & \frac{\partial^2 C(\boldsymbol{p})}{\partial p_1 \partial p_3} & \frac{\partial^2 C(\boldsymbol{p})}{\partial p_1 \partial p_4} & \frac{\partial^2 C(\boldsymbol{p})}{\partial p_1 \partial p_5} & \frac{\partial^2 C(\boldsymbol{p})}{\partial p_1 \partial p_6} \\ \frac{\partial^2 C(\boldsymbol{p})}{\partial p_2 \partial p_1} & \frac{\partial^2 C(\boldsymbol{p})}{\partial p_2^2} & \frac{\partial^2 C(\boldsymbol{p})}{\partial p_2 \partial p_3} & \frac{\partial^2 C(\boldsymbol{p})}{\partial p_2 \partial p_4} & \frac{\partial^2 C(\boldsymbol{p})}{\partial p_2 \partial p_5} & \frac{\partial^2 C(\boldsymbol{p})}{\partial p_2 \partial p_6} \\ \frac{\partial^2 C(\boldsymbol{p})}{\partial p_3 \partial p_1} & \frac{\partial^2 C(\boldsymbol{p})}{\partial p_3 \partial p_2} & \frac{\partial^2 C(\boldsymbol{p})}{\partial p_3^2} & \frac{\partial^2 C(\boldsymbol{p})}{\partial p_3 \partial p_4} & \frac{\partial^2 C(\boldsymbol{p})}{\partial p_3 \partial p_5} & \frac{\partial^2 C(\boldsymbol{p})}{\partial p_3 \partial p_6} \\ \frac{\partial^2 C(\boldsymbol{p})}{\partial p_4 \partial p_1} & \frac{\partial^2 C(\boldsymbol{p})}{\partial p_4 \partial p_2} & \frac{\partial^2 C(\boldsymbol{p})}{\partial p_4 \partial p_3} & \frac{\partial^2 C(\boldsymbol{p})}{\partial p_4^2} & \frac{\partial^2 C(\boldsymbol{p})}{\partial p_4 \partial p_5} & \frac{\partial^2 C(\boldsymbol{p})}{\partial p_4 \partial p_6} \\ \frac{\partial^2 C(\boldsymbol{p})}{\partial p_5 \partial p_1} & \frac{\partial^2 C(\boldsymbol{p})}{\partial p_5 \partial p_2} & \frac{\partial^2 C(\boldsymbol{p})}{\partial p_5 \partial p_3} & \frac{\partial^2 C(\boldsymbol{p})}{\partial p_5 \partial p_4} & \frac{\partial^2 C(\boldsymbol{p})}{\partial p_5^2} & \frac{\partial^2 C(\boldsymbol{p})}{\partial p_5 \partial p_6} \\ \frac{\partial^2 C(\boldsymbol{p})}{\partial p_6 \partial p_1} & \frac{\partial^2 C(\boldsymbol{p})}{\partial p_6 \partial p_2} & \frac{\partial^2 C(\boldsymbol{p})}{\partial p_6 \partial p_3} & \frac{\partial^2 C(\boldsymbol{p})}{\partial p_6 \partial p_4} & \frac{\partial^2 C(\boldsymbol{p})}{\partial p_6 \partial p_5} & \frac{\partial^2 C(\boldsymbol{p})}{\partial p_6^2} \end{bmatrix}. \tag{3.7}$$

As such Equation 3.5 can be rewritten in a more intuitive form as

$$\boldsymbol{p}_{i+1} = \boldsymbol{p}_i + \boldsymbol{\Delta_p} \tag{3.8}$$
$$\text{where} \quad \boldsymbol{H}\boldsymbol{\Delta_p} = -\boldsymbol{J}. \tag{3.9}$$

This is done because the division of the Jacobian by the Hessian in Equation 3.5 is not a true representation of the mathematical operations involved. Here $\boldsymbol{\Delta_p}$ is the change in the WFPs that improves the current estimate. This vector is determined by solving the system of linear equations given in Equation 3.9.

3.1.1 Modified Newton-Raphson

Ideally Equation 3.8 would determine the WFPs, in an iterative manner, that minimise the correlation criterion therefore solving the correspondence problem. However the Newton-Raphson method becomes less reliable as the number of independent dimensions, the WFPs, increase. The modified Newton-Raphson method accounts for this by combining the Newton-Raphson method with a line search method.

cite here? increase vs increases

Consider Equation 3.8. This equation can be interpreted as improving the current estimate, $\boldsymbol{p}_i$, by stepping a distance of one along the search direction, $\boldsymbol{\Delta_p}$. Using this interpretation the Newton-Raphson method can be made more robust by using the golden section line search method to determine the optimal step size along the search direction. Thus Equation 3.8 becomes

$$\boldsymbol{p}_{i+1} = \boldsymbol{p}_i + \alpha_{opt}\boldsymbol{\Delta_p} \tag{3.10}$$

where α_{opt} is the optimal step size determined using the golden section line search method and $\boldsymbol{\Delta_p}$ is the search direction determined by the Newton-Raphson method.

The golden section method is a bracketing method which takes in a lower and upper limit for the step size and progressively moves these limits closer to one another such that the optimal step size is isolated between the two limits. Once the range spanned between the two limits is smaller than a predefined tolerance then the optimal step size is taken as the average of the two limits.

Within each iteration of the golden section method two step sizes are investigated. Letting α_l and α_u be the lower and upper limits of the step size respectively the two investigated step sizes are

$$\alpha_1 = \alpha_l + (1 - \varphi) \times (\alpha_u - \alpha_l) \tag{3.11}$$

$$\text{and} \quad \alpha_2 = \alpha_l + \varphi \times (\alpha_u - \alpha_l) \tag{3.12}$$

$$\text{where} \quad \varphi = \frac{\sqrt{(}5) - 1}{2}. \tag{3.13}$$

The correlation criterion is then evaluated for these two step sizes.

$$C_1 = C(\boldsymbol{p}_i + \alpha_1 \times \boldsymbol{\Delta_p}) \quad \text{and} \quad C_2 = C(\boldsymbol{p}_i + \alpha_2 \times \boldsymbol{\Delta_p}) \tag{3.14}$$

If C_1 is less than C_2 then the upper limit changes to α_2.

Thus the correlation criterion is evaluated as

$$C_2 = C(\boldsymbol{p}_i + \alpha_2 \times \boldsymbol{\delta_p}) \tag{3.15}$$

$$C_3 = C(\boldsymbol{p}_i + \alpha_3 \times \boldsymbol{\delta_p}) \tag{3.16}$$

Within each iteration of the golden section method one of the limits is changed such that it lies closer to the unchanged one. This is done by

the correlation coefficient is evaluated for two step sizes that lie within the limits and the dslfasdj fadl;k

The golden section method is unique in that

limits bracketing golden ratio - why stopping condition

The golden section method takes in a lower and upper limit for the step size and determines the optimal step size that lies within this range. It does this by bracketing the range spanned by these limits such that the optimum lies within this bracketed segment.

The golden section method is a bracketing search method which

The golden section method determines the optimal step size within an allowable step-size range defined by the lower and upper limits represented as α_0 and α_1 respectively. The correlation criterion is assumed to be unimodal over this range and so the golden section method

It does this by reducing the step-size range by determining the segment of the range within which the optimum is contained.

The golden section search method is a bracketing search method. This means that it determines the optimal step size, within an allowable step-size range, that optimises the correlation criterion in a certain search direction. The step-size range is defined by the lower and upper limits placed on the step size which are indicated as α_0 and α_1 respectively.

Within the first iteration of the method the correlation criterion is evaluated at two step sizes that fall within the step size range; namely α_2 and α_3.

The magnitude of these step sizes are determined by the golden ration, φ, and the limits place on the step size.

$$\alpha_2 = \alpha_0 + (1-\varphi) \times (\alpha_1 - \alpha_0) \tag{3.17}$$

$$\alpha_3 = \alpha_0 + \varphi \times (\alpha_1 - \alpha_0) \tag{3.18}$$

$$\text{where} \quad \varphi = \frac{\sqrt{(5)} - 1}{2} \tag{3.19}$$

Thus the correlation criterion is evaluated as

$$C_2 = C(\boldsymbol{p}_i + \alpha_2 \times \boldsymbol{\delta_p}) \tag{3.20}$$

$$C_3 = C(\boldsymbol{p}_i + \alpha_3 \times \boldsymbol{\delta_p}) \tag{3.21}$$

where $\boldsymbol{p}_i$ is the previous estimate of the WFPs.

Once the correlation criterion is evaluated for these two step sizes the correlation coefficients are compared. If C_2 is less than C_3 then it is known that the minimum of the correlation criterion lies between step sizes α_0 and α_3. Otherwise, if C_2 is greater than C_3, the minimum lies between step sizes α_2 and α_1. The latter is illustrated in Figure 3.1. Thus on the next iteration of the golden section method the step-size search interval changes based on the conditions above thereby shrinking the search interval.

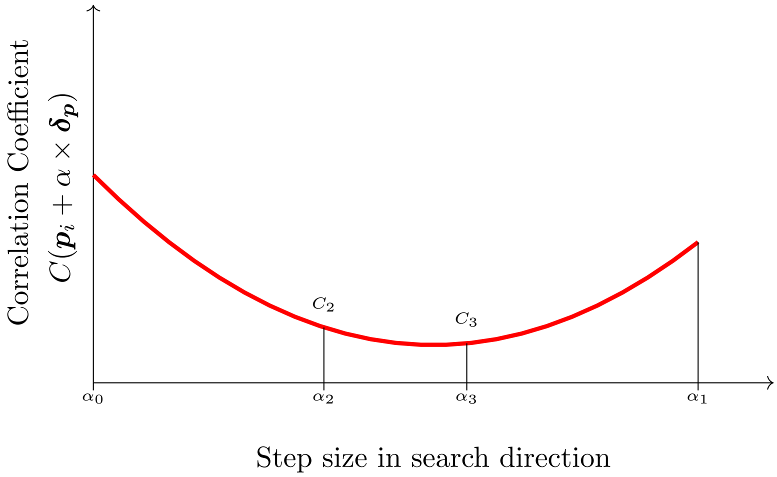


Figure 3.1: First iteration of golden section search method

Within the second iteration the process is repeated to find two new step sizes to investigate within the new step-size search interval, between α_2 and α_1. These two new step sizes are determined in the same way as in Equations 3.17 and 3.18 but with α_2 and α_1 acting as the new lower and upper limits.

$$\alpha_4 = \alpha_2 + (1-\varphi) \times (\alpha_1 - \alpha_2) \tag{3.22}$$

$$\alpha_5 = \alpha_2 + \varphi \times (\alpha_1 - \alpha_2) \tag{3.23}$$

It can be shown that α_4 is equivalent to α_3 by substituting Equation 3.17 for α_2 in Equation 3.22 and noting that $\varphi^2 = 1 - \varphi$.

$$\alpha_4 = \alpha_0 + (1-\varphi) \times (\alpha_1 - \alpha_0) + (1-\varphi) \times (\alpha_1 - (\alpha_0 + (1-\varphi) \times (\alpha_1 - \alpha_0))) \tag{3.24}$$

$$= \alpha_1 + \varphi^2(\alpha_0 - \alpha_1) \tag{3.25}$$

$$= \alpha_1 + (1-\varphi)(\alpha_0 - \alpha_1) \tag{3.26}$$

$$= \alpha_0 + \varphi(\alpha_1 - \alpha_0) = \alpha_3 \tag{3.27}$$

Thus since α_4 is equivalent to α_3 the correlation coefficient C_4 need not be calculated since it is equivalent to C_3. This can be seen when comparing Figures 3.1 and 3.2. This phenomenon, where one of the new step sizes is equivalent to a step size in the previous iteration, occurs during each subsequent iteration of the method after the first iteration. This is a property of the golden ratio and it is the reason for using it to determine the step sizes. The benefit of this is that the correlation criterion must be evaluated only once per iteration after the first iteration which is significant since evaluating the correlation criterion is computationally expensive.

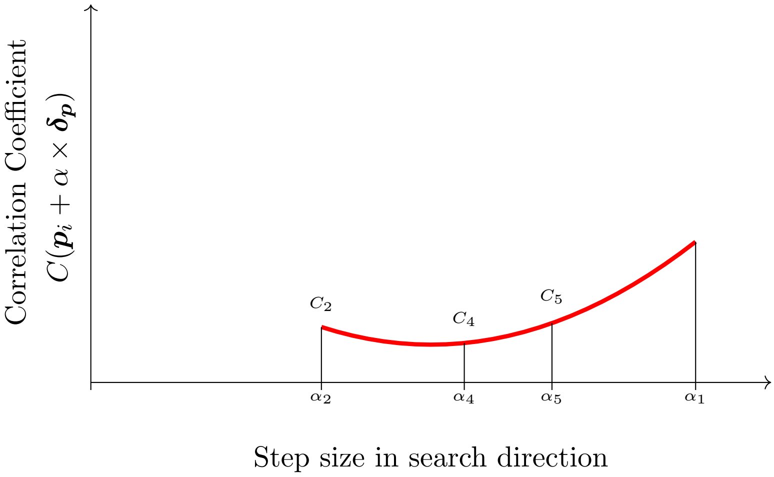


Figure 3.2: Second iteration of golden section search method

The golden section method continues to iterate in this manner to find an optimal step size. The iterations stop once an iteration is reached where the distance between the two step sizes, where the correlation criterion is to be evaluated, is smaller than a predefined tolerance. At this point the optimal step size is taken to be the average of these last two step sizes.

talk about memory requirements and interpolation requirements, concerns of accuracy of initial

3.1.2 Computing the Jacobian and Hessian

Although the Jacobian and Hessian can be determined using numerical approximation it was decided to determine an analytical expression for them. The reason for this is to that the similarities between this method and the Gauss-Newton method can be made clear. The differentiation of the Jacobian and Hessian is performed in two steps. First the chosen correlation criterion is differentiated according to Equations 3.6 and 3.7 up until the point that it is the light intensity values of the deformed subset that are being differentiated. To illustrate this consider the Zero-Mean Normalised Sum of Squared Difference correlation criterion of Equation 2.107. The j-th element of the Jacobian for this correlation criterion is given as

$$J_j = \frac{\partial C_{ZNSSD}}{\partial p_j} = 2\sum_i^n \left[\frac{F_i - \bar{F}}{\sqrt{\sum_i^n (F_i - \bar{F})^2}} - \frac{G(\boldsymbol{x}_0 + W(\boldsymbol{\zeta}_i, \boldsymbol{p})) - \bar{G}}{\sqrt{\sum_i^n (G_i - \bar{G})^2}} \right] \times \left(- \frac{\frac{\partial G(\boldsymbol{x}_0 + W(\boldsymbol{\zeta}_i, \boldsymbol{p}))}{\partial p_j}}{\sqrt{\sum_i^n (G_i - \bar{G})^2}} \right). \tag{3.28}$$

An element of the accompanying Hessian is then

$$\begin{aligned} H_{jk} = \frac{\partial^2 C_{ZNSSD}}{\partial p_j \partial p_k} &= 2\sum_i^n \left[\frac{F_i - \bar{F}}{\sqrt{\sum_i^n (F_i - \bar{F})^2}} - \frac{G(\boldsymbol{x}_0 + W(\boldsymbol{\zeta}_i, \boldsymbol{p})) - \bar{G}}{\sqrt{\sum_i^n (G_i - \bar{G})^2}} \right] \times \left(- \frac{\frac{\partial^2 G(\boldsymbol{x}_0 + W(\boldsymbol{\zeta}_i, \boldsymbol{p}))}{\partial p_j \partial p_k}}{\sqrt{\sum_i^n (G_i - \bar{G})^2}} \right) \\ &+ 2\sum_i^n \left[\left(- \frac{\frac{\partial G(\boldsymbol{x}_0 + W(\boldsymbol{\zeta}_i, \boldsymbol{p}))}{\partial p_j}}{\sqrt{\sum_i^n (G_i - \bar{G})^2}} \right) \times \left(- \frac{\frac{\partial G(\boldsymbol{x}_0 + W(\boldsymbol{\zeta}_i, \boldsymbol{p}))}{\partial p_k}}{\sqrt{\sum_i^n (G_i - \bar{G})^2}} \right) \right]. \end{aligned} \tag{3.29}$$

These equations are derived by hand and then coded into a function which determines the Jacobian and Hessian as a function of the light intensities and derivatives of G.

To complete the expression for the Jacobian and Hessian the partial derivatives of the light intensities of the deformed subset in terms of the WFPs need to be determined. This is the second step. This differentiation is achieved by differentiating the interpolation equation, that has been fit to the light intensities of the deformed subset, in terms of the WFPs. This is done using Matlabs symbolic toolbox which can determine derivatives of symbolic expressions.

The code for determining the derivatives of G is given in Figure 3.3. First symbolic variables for the interpolation coefficients, WFPs, subset centre position and distance from the subset centre to the pixel under consideration are declared (lines 2-4). Then the warp function of Equation 2.96 is used to define the query points in terms of these symbolic variables (line 7-8). Thereafter the fractional parts of the query point are calculated (line 10-11). Finally the interpolation equation, Equation 2.114, is defined (line 13). This equation give the interpolation equation as a function of all the variables defined in lines 2-4.

With the interpolation equation as a function of the WFPs this equation can be symbolically differentiated in terms of the WFPs. This is done in lines

```
1  % Declare the necessary variables symbolically
2  a = sym('a_%d', [4 4]) % The coefficients of the ...
       interpolation equation
3  P = sym('p_%d', [6,1]) % The warp function parameters
4  syms dx dy x0 y0; % dx = ∆ x, dy = ∆ y and (x0,y0) is the ...
       centre of the subset
5
6  % Use the warp function to determine the warped coordinates...
        (query points)
7  x_q=P(1)+P(3).*dy+dx.*(P(2)+1.0)+x0;
8  y_q=P(4)+P(5).*dx+dy.*(P(6)+1.0)+y0;
9  % Determine the fractional part of the query points
10 x_fq=x_q-floor(x_q);
11 y_fq=y_q-floor(y_q);
12 % Create the function for interpolating the query points - ...
       this equation is a function of the warp function ...
       parameters, the interpolation coefficients, subset ...
       centre and pixel position
13 G_interp=[1, x_fq, x_fq^2, x_fq^3]*a*[1; y_fq; y_fq^2; y_fq...
       ^3];
14 % Calculate the Jacobian and Hessian of the interpolation ...
       equation
15 J_G=jacobian(G_interp,P);
16 H_G=hessian(G_interp,P);
17 % Convert these symbolic expressions of the Jacobian and ...
       Hessian into matlab functions which are dependent on the...
        necessary variables.
18 matlabFunction(J_G,'File','JacobianFunction','Optimize',...
       true,'Vars',{a,P,dx,dy,x0,y0});
19 matlabFunction(H_G,'File','HessianFunction','Optimize',true...
       ,'Vars',{a,P,dx,dy,x0,y0});
```

Figure 3.3: Code for determining the derivatives of the interpolation equation in terms of the WFPs (step 2)

15 and 16 where the Jacobian and Hessian of G_{interp}, in terms of the WFPs, are determined. Matlab's Jacobian and Hessian functions are used because they determine all the necessary derivatives of G_{interp} in terms of the WFPs. The derivatives are then saved as functions of the variables defined in lines 2-4 such that these functions can be used to evaluate $\frac{\partial G}{\partial p_j}$ and $\frac{\partial^2 G}{\partial p_j \partial p_k}$ for any estimate of the WFPs, in any image and for any subset within the image.

Once the code for the second step has completed running two functions are available for calculating $\frac{\partial G}{\partial p_j}$ and $\frac{\partial^2 G}{\partial p_j \partial p_k}$ namely JacobianFunction and HessianFunction. These can now be incorporated into the code written for Equations 3.28 and 3.29 in the first step. This code is shown in Figure 3.4.

The reason for separating the differentiation into two steps is because the

```
1  % Here F is the reference subset light intensity values and ...
       G is the interpolated values of the deformed subset
2  Fmean=mean(mean(F));
3  Gmean=mean(mean(G));
4  Fsqrt=sqrt(sum(sum((F-Fmean).^2)));
5  Gsqrt=sqrt(sum(sum((G-Gmean).^2)));
6  % initialise Jacobian and Hessian
7  J=zeros([numP,1]);
8  H=zeros([numP,numP]);
9  for i=1:n % summation over whole subset (n=number of pixels)
10  % determine derivatives of light intensity values
11  J_G=(JacobianFunction(coef{i},P',dx(i),dy(i),X,Y));
12  H_G=(HessianFunction(coef{i},P',dx(i),dy(i),X,Y));
13  % use derivatives to compute Jacobian and Hessian
14  J=J+((F(i)-Fmean)/Fsqrt-(G(i)-Gmean)/Gsqrt).*((-J_G')./...
        Gsqrt);
15  H=H+(-J_G'./Gsqrt)*(-J_G./Gsqrt) + ((F(i)-Fmean)/Fsqrt-(G(i...
        )-Gmean)/Gsqrt).*(-H_G)./Gsqrt; %if transpose jacky ...
        outside of brackets it doesn't work
16 end
17 J=2*J;
18 H=2*H;
```

Figure 3.4: Code for determining the Jacobian and Hessian according to Equations 3.28 and 3.29 (step 1)

first step is dependent on the correlation criterion but independent of the warp function. In contrast the second step is independent of the correlation criterion but dependent on the warp function.

Therefore if a different correlation coefficient is used only the code on lines 14 and 15 of Figure 3.4 need to be changed. Similarly if a different warp function is used only the code on lines 3, 7 and 8 in Figure 3.3 need to be changed. This is done automatically based on the correlation criterion chosen and the warp function defined by the user.

explain that whole jac symbolically would be expensive and change often

3.2 Gauss-Newton

The Gauss-Newton algorithm is another popular method of solving the correspondence problem. Fundamentally it makes use of the Gauss-Newton gradient descent non-linear optimisation algorithm to minimise the correlation criterion [6]. The Gauss-Newton method is a modification of the Newton-Raphson method in which the Hessian is calculated from first derivatives. As such it avoids calculating second derivatives which are computationally expensive.

Instead of optimising the correlation criterion in terms of the current estimate of the WFPs the Gauss-Newton algorithm optimises it in terms of the iterative improvement of the WFPs. The Gauss-Newton method can be implemented in different ways which differ in how the iterative improvement of the WFPs, $\Delta\boldsymbol{p}$, is applied within the correlation criterion. These different implementations result in different algorithms. Two of these implementations are presented in the following sections. The first is the forward additive implementation which is the original form known as the Lucas-Kanade algorithm [18]. The second is the inverse compositional implementation which offers greater computational efficiency.

The algorithms derived in this section are based on work done by Baker [6] and Pan [22].

3.2.1 Forward additive

For the forward additive algorithm the WFPs, $\boldsymbol{p}$, and the iterative improvement of the WFPs, $\Delta\boldsymbol{p}$, are both applied to the investigated subset, G. This corresponds to how they were applied in the Newton-Raphson method. Thus by applying this to the Zero-Mean Normalised Sum of Squared Difference correlation criterion results in

show this

$$C_{ZNSSD} = \sum_{i}^{n} \left[\frac{F(\boldsymbol{x}_0 + \boldsymbol{\zeta}_i) - \bar{F}}{\sqrt{\sum_i^n (F_i - \bar{F})^2}} - \frac{G(\boldsymbol{x}_0 + W(\boldsymbol{\zeta}_i, \boldsymbol{p} + \Delta\boldsymbol{p})) - \bar{G}}{\sqrt{\sum_i^n (G_i - \bar{G})^2}} \right]^2 \tag{3.30}$$

Taking the first-order Taylor series expansion of Equation 3.30 in terms of $\Delta\boldsymbol{p}$ gives

first order with hyphen ?

$$C_{ZNSSD} = \sum_{i} \left[\frac{F(\boldsymbol{x}_0 + \boldsymbol{\zeta}_i) - \bar{F}}{\sqrt{\sum_i^n (F_i - \bar{F})^2}} - \frac{G(\boldsymbol{x}_0 + W(\boldsymbol{\zeta}_i, \boldsymbol{p})) + \nabla G_i \frac{\partial W_i}{\partial \boldsymbol{p}} \Delta\boldsymbol{p} - \bar{G}}{\sqrt{\sum_i^n (G_i - \bar{G})^2}} \right]^2 \tag{3.31}$$

$$\text{where} \quad \nabla G_i = \begin{bmatrix} \frac{\partial G(\boldsymbol{x}_0 + \boldsymbol{\zeta}_i)}{\partial x} & \frac{\partial G(\boldsymbol{x}_0 + \boldsymbol{\zeta}_i)}{\partial y} \end{bmatrix} \tag{3.32}$$

$$\text{and} \quad \frac{\partial W_i}{\partial \boldsymbol{p}} = \begin{bmatrix} 1 & \Delta x & \Delta y & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & \Delta x & \Delta y \end{bmatrix}. \tag{3.33}$$

Here ∇G_i is the light intensity gradients of the investigated subset in the x and y directions for pixel i and $\frac{\partial W}{\partial \boldsymbol{p}}$ is the Jacobian of the warp function for pixel i. Making use of the Taylor series expansion in this way eliminates the need to differentiate the interpolation algorithm.

In other words the investigated subset does not need to be warped, and therefore interpolated, before calculating its light intensity gradients as done in the Newton-Raphson method. Instead the light intensity gradients of the investigated subset are calculated, with no deformation applied, and then these gradients are warped according to the current estimate of the WFPs using term $\frac{\partial W_i}{\partial \boldsymbol{p}}\Delta\boldsymbol{p}$.

Taking the derivative of equation 3.31 with respect to $\Delta\boldsymbol{p}$ and setting it to zero gives the least squares solution for finding $\Delta\boldsymbol{p}$.

$$\begin{aligned}\frac{\partial C_{ZNSSD}}{\partial \Delta\boldsymbol{p}} &= 0 \\ &= 2\sum_i \frac{-\left(\nabla G_i \frac{\partial W_i}{\partial \boldsymbol{p}}\right)^T}{\sqrt{\sum_i^n (G_i-\bar{G})^2}}\left[\frac{F(\boldsymbol{x}_0+\boldsymbol{\zeta}_i)-\bar{F}}{\sqrt{\sum_i^n (F_i-\bar{F})^2}} - \frac{G(\boldsymbol{x}_0+W(\boldsymbol{\zeta}_i,\boldsymbol{p}))+\nabla G_i \frac{\partial W_i}{\partial \boldsymbol{p}}\Delta\boldsymbol{p}-\bar{G}}{\sqrt{\sum_i^n (G_i-\bar{G})^2}}\right]\end{aligned} \tag{3.34}$$

$$\Delta\boldsymbol{p} = -\boldsymbol{H}^{-1}\times\sum_i\left[\frac{-\left(\nabla G_i \frac{\partial W_i}{\partial \boldsymbol{p}}\right)^T}{\sqrt{\sum_i^n (G_i-\bar{G})^2}}\times\left[\frac{F(\boldsymbol{x}_0+\boldsymbol{\zeta}_i)-\bar{F}}{\sqrt{\sum_i^n (F_i-\bar{F})^2}} - \frac{G(\boldsymbol{x}_0+W(\boldsymbol{\zeta}_i,\boldsymbol{p}))-\bar{G}}{\sqrt{\sum_i^n (G_i-\bar{G})^2}}\right]\right] \tag{3.35}$$

$$\text{where}\quad \boldsymbol{H} = \sum_i\left[\frac{-(\nabla G_i \frac{\partial W_i}{\partial \boldsymbol{p}})^T}{\sqrt{\sum_i^n (G_i-\bar{G})^2}}\frac{-(\nabla G_i \frac{\partial W_i}{\partial \boldsymbol{p}})}{\sqrt{\sum_i^n (G_i-\bar{G})^2}}\right] \tag{3.36}$$

Here the first term on the RHS of Equation 3.35 is the Hessian, $\boldsymbol{H}$, and the second term is the Jacobian for the Gauss-Newton method. Keeping in mind that $\nabla G_i \frac{\partial W}{\partial \boldsymbol{p}}$ is the linear approximation of $\frac{\partial G(\boldsymbol{x}_0+W(\zeta,\boldsymbol{p}))}{\partial \boldsymbol{p}}$ it becomes clear that the Gauss-Newton Jacobian of Equation 3.35 is directly related to the Newton-Raphson Jacobian of Equation 3.28.

Similarly when considering the Hessian in Equation 3.36 and the Newton-Raphson Hessian in Equation 3.28 it is clear that the first term of Equation 3.28 is dropped in the Gauss-Newton method. This is referred to as the Guass-Newton approximation of the Hessian in which second derivatives are ignored.

Therefore it can be seen that the Lucas-Kanade algorithm and the Newton-Raphson algorithm are closely related with only two major differences between them. The first difference is the linear approximation applied to $\frac{\partial G(\boldsymbol{x}_0+W(\zeta,\boldsymbol{p}))}{\partial \boldsymbol{p}}$ which simplifies the process of finding the derivatives and makes the derivative independent of the interpolation method. Secondly the Gauss-Newton approximation of the Hessian in which second derivatives are ignored. These simplifications greatly reduce the computation cost of solving for $\Delta\boldsymbol{p}$ when compared to the Newton-Raphson method.

According to how $\Delta\boldsymbol{p}$ was applied in Equation 3.30 the current estimate of the WFPs, $\boldsymbol{p}_j$, is related to the improved estimate, $\boldsymbol{p}_{j+1}$, by

$$\boldsymbol{p}_{j+1} = \boldsymbol{p}_j + \Delta\boldsymbol{p} \tag{3.37}$$

3.2.2 Inverse compositional

Although the method presented in Section 3.2.1 offers reduced computational cost per iteration when compared to the Newton-Raphson method, its efficiency can be improved further by changing the way in which the iterative update of the WFPs is applied within the correlation criterion.

The algorithm explained here is based on the one proposed by B. Pan [22]. It is the inverse compositional Gauss-Newton method. The current guess of the WFPs, $\boldsymbol{p}$, is applied to the investigated subset, G, while the iterative improvement of the WFPs, $\Delta\boldsymbol{p}$, is applied to the reference subset, F, as shown below.

$$C_{ZNSSD} = \sum_{i}^{n} \left[\frac{F(\boldsymbol{x_0} + W(\boldsymbol{\zeta}_i, \Delta\boldsymbol{p})) - \bar{F}}{\sqrt{\sum_i^n (F_i - \bar{F})^2}} - \frac{G(\boldsymbol{x_0} + W(\boldsymbol{\zeta}_i, \boldsymbol{p})) - \bar{G}}{\sqrt{\sum_i^n (G_i - \bar{G})^2}} \right]^2 \tag{3.38}$$

Taking the first-order Taylor expansion of equation 3.38 in terms of $\Delta\boldsymbol{p}$ gives

$$C_{ZNSSD} = \sum_{i} \left[\frac{F(\boldsymbol{x_0} + \boldsymbol{\zeta}_i) + \nabla F \frac{\partial W}{\partial \boldsymbol{p}} \Delta\boldsymbol{p} - \bar{F}}{\sqrt{\sum_i^n (F_i - \bar{F})^2}} - \frac{G(\boldsymbol{x_0} + W(\boldsymbol{\zeta}_i, \boldsymbol{p})) - \bar{G}}{\sqrt{\sum_i^n (G_i - \bar{G})^2}} \right]^2 \tag{3.39}$$

$$\text{where} \quad \nabla F = \begin{bmatrix} \frac{\partial F(\boldsymbol{x_0} + \boldsymbol{\zeta})}{\partial x} & \frac{\partial F(\boldsymbol{x_0} + \boldsymbol{\zeta})}{\partial y} \end{bmatrix}. \tag{3.40}$$

Here ∇F is the light intensity gradients of the reference subset in the x and y directions and $\frac{\partial W}{\partial \boldsymbol{p}}$ is the Jacobian of the warp function which remains the same as in Equation 3.33.

Taking the derivative of Equation 3.39 with respect to $\Delta\boldsymbol{p}$ and setting it to zero gives the least squares solution for finding $\Delta\boldsymbol{p}$.

$$\begin{aligned} \frac{\partial C_{ZNSSD}}{\partial \Delta\boldsymbol{p}} &= 0 \\ &= 2 \sum_{i} \frac{-(\nabla F_i \frac{\partial W_i}{\partial \boldsymbol{p}})^T}{\sqrt{\sum_i^n (F_i - \bar{F})^2}} \left[\frac{F(\boldsymbol{x}_0 + \boldsymbol{\zeta}_i) + \nabla F_i \frac{\partial W_i}{\partial \boldsymbol{p}} \Delta\boldsymbol{p} - \bar{F}}{\sqrt{\sum_i^n (F_i - \bar{F})^2}} - \frac{G(\boldsymbol{x}_0 + W(\boldsymbol{\zeta}_i, \boldsymbol{p})) - \bar{G}}{\sqrt{\sum_i^n (G_i - \bar{G})^2}} \right] \end{aligned} \tag{3.41}$$

$$\Delta\boldsymbol{p} = -\boldsymbol{H}^{-1} \times \sum_{i} \left[\frac{-(\nabla F_i \frac{\partial W_i}{\partial \boldsymbol{p}})^T}{\sqrt{\sum_i^n (F_i - \bar{F})^2}} \times \left[\frac{F(\boldsymbol{x}_0 + \boldsymbol{\zeta}_i) - \bar{F}}{\sqrt{\sum_i^n (F_i - \bar{F})^2}} - \frac{G(\boldsymbol{x}_0 + W(\boldsymbol{\zeta}_i, \boldsymbol{p})) - \bar{G}}{\sqrt{\sum_i^n (G_i - \bar{G})^2}} \right] \right] \tag{3.42}$$

$$\text{where} \qquad \boldsymbol{H} = \sum_{i} \left[\frac{-(\nabla F_i \frac{\partial W_i}{\partial \boldsymbol{p}})^T}{\sqrt{\sum_i^n (F_i - \bar{F})^2}} \frac{-(\nabla F_i \frac{\partial W_i}{\partial \boldsymbol{p}})}{\sqrt{\sum_i^n (F_i - \bar{F})^2}} \right] \tag{3.43}$$

Comparing Equations 3.42 and 3.43 to Equations 3.35 and 3.36 it is clear that the only difference is that $\frac{-(\nabla G_i \frac{\partial W_i}{\partial \boldsymbol{p}})}{\sqrt{\sum_i^n (G_i - \bar{G})^2}}$ has been replaced with $\frac{-(\nabla F_i \frac{\partial W_i}{\partial \boldsymbol{p}})}{\sqrt{\sum_i^n (F_i - \bar{F})^2}}$.

This is significant because this term in the forward additive method changes on each iteration of the algorithm because it is dependent on the WFPs. However the term in the inverse compositional algorithm is independent of the WFPs and thus remains unchanged throughout the whole correlation process. Therefore it can be calculated before starting the iterations which significantly reduces the computational cost of each iteration.

Furthermore Baker and Matthews [6] proved that the forward additive algorithm is equivalent to the inverse compositional algorithm. Thus the inverse compositional algorithm offers greater efficiency without compromising accuracy. It is for this reason that this algorithm is focused on in this project.

However in order to achieve this reduced computational cost the iterative improvement to the WFPs was applied to the reference subset. This means that the iterative improvement cannot simply be added to the current estimate of the WFPs to obtain the updated WFPs as in the forward additive method. Instead the updated WFPs must be determined using matrix operations. Defining the function $W_{mat}(\boldsymbol{p})$ which arranges the WFPs in a matrix the updated WFPs, $\boldsymbol{p}_{j+1}$, can be determined as

$$W_{mat}(\boldsymbol{p}) = \begin{bmatrix} 1+p_2 & p_3 & p_1 \\ p_5 & p_6+1 & p_4 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.44}$$

$$W_{mat}(\boldsymbol{p}_{j+1}) = W_{mat}(\boldsymbol{p}_j) \times W_{mat}(\Delta\boldsymbol{p})^{-1}. \tag{3.45}$$

The new approximation to the WFPs can then be extracted from this matrix, $W_{mat}(\boldsymbol{p}_{j+1})$, and used in the next iteration of the algorithm.

3.2.3 Subset splitting

One of the major downfalls of subset based DIC is that it assumes that the deformation field over every subset is continuous. Although this is often the case it breaks down when there is a displacement discontinuity in the displacement field of the specimen resulting from the presence of cracks or shear bands. In order to deal with this issue the method of subset splitting proposed by Poissant and Barthelat [25] was implemented. In this method an equation defining a line, that passes through the subset, is used to split the pixels within the subset into two groups and a different set of WFPs are used for each group. This is illustrated in Figure 3.5. Thus if the line is placed such that it lies on the crack the subset will be split into two groups allowing the deformations on either side of the crack to be determined separately.

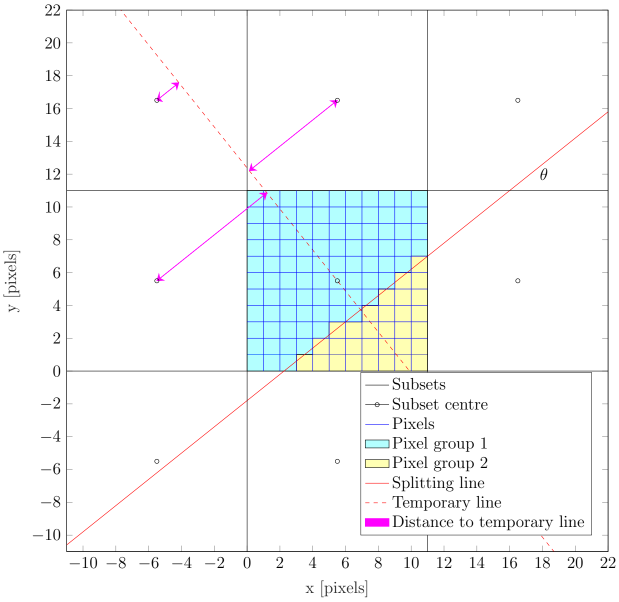


Figure 3.5: How the splitting line divides the pixel of the subset into two groups

The first step is to determine the equation for the line such that it lies on the crack. A straight line is used to approximate the crack because a curved line would make the process more complex and is in general not necessary since the subsets only capture a small portion of the crack. The method proposed by Poissant and Barthelat [25] for determining the equation of this line was found to give inconsistent results. Instead a more robust method, although more computationally expensive, was implemented in the form of particle swarm optimisation. The equation for the line is of the form

$$y = m_{grad} \times (x - x_{shift}) + y_{shift} \tag{3.46}$$

$$\text{where} \quad m_{grad} = \tan\theta \tag{3.47}$$

where x_{shift} is the shift in the x direction, y_{shift} is the shift in the y direction and m_{grad} is the gradient which corresponds with angle θ that the line makes with the x-axis. Note that the origin of the coordinate system is placed at the bottom left corner of the subset. The reason for including a shift in both directions is so that a definitive bounds can be placed on the range of values evaluated by the particle swarm algorithm. In other words if the shift in the x direction, x_{shift}, was taken out of the equation then the shift in the y direction, y_{shift}, would have to range from negative infinity to infinity in order to compensate. Within the particle swarm optimisation the x and y directional shifts are evaluated from zero to the subset size and the angle θ is evaluated from 0° to 180° measured anti-clockwise from the x-axis.

Particle swarm optimisation is performed using the correlation criterion as the objective function. However the investigated subset first needs to be deformed before the correlation criterion can be evaluated. The particle swarm algorithm first guesses values for variables m_{grad}, y_{shift} and x_{shift} which defines a splitting line. The pixels within the investigated subset are split into two groups according to this splitting line as shown in Figure 3.5. A different set of WFPs needs to be applied to each of these groups of pixels to obtain light intensities for the deformed subset so that the correlation criterion can be evaluated.

These two sets of WFPs are taken as the WFPs of two subsets, with good correlation coefficient values, nearest to the subset under consideration. To find these two subsets a temporary line is drawn through the centre of the investigated subset such that this line is perpendicular to the splitting line. Then for each neighbouring subset the shortest distance from its centre to this temporary line is determined. A neighbouring subset is defined as a subset that is within a reasonable distance of the investigated subset.

The neighbouring subsets are split into two groups based on which side of the splitting line they reside. The two subsets with the shortest distance, which also have good correlation coefficients, are then identified from these groups. The WFPs of these subsets are then used to warp the two groups of pixels within the investigated subset in order to obtain the query points for interpolation. Interpolation is performed to obtain a light intensity matrix for the deformed subset which is used in the correlation criterion to determine the degree of fit for the splitting line considered. In this way the particle swarm optimiser considers many different splitting lines and finds the splitting line that best fits the subset.

The second step is to perform correlation on the investigated subset according to the splitting line found from particle swarm optimisation. This requires minimising the correlation criterion of the subset by optimising two sets of WFPs each of which is applied to a separate group of pixels that lie on either side of the splitting line. To do this the inverse compositional Gauss-Newton algorithm discussed earlier needs to be modified slightly.

First the line equation is used to split the pixels within the subset into these two groups based on which side of the line the pixels are located. Then each group of pixels are warped according to the WFPs that are applicable to the group. This results in the query points for the subset. The deformed image is interpolated at these locations to obtain $G(\boldsymbol{x}_0+W(\boldsymbol{\zeta}_i,\boldsymbol{p}))$. Thereafter Equation 3.42 is evaluated separately for each group of pixels. In other words the summations of Equations 3.42 and 3.43 are limited to pixels that fall within the same group.

This results in a $\Delta\boldsymbol{p}$ for each set of WFPs. Then Equation 3.45 is applied to each set of WFPs separately using their respective $\Delta\boldsymbol{p}$ to obtain an improved estimate of the WFPs. Thereafter $G(\boldsymbol{x}_0 + W(\boldsymbol{\zeta}_i, \boldsymbol{p}))$ is recalculated in the same way using the updated sets of WFPs. This is then used to determine the correlation coefficient of the whole subset subjected to the updated WFPs to determine whether correlation has been achieved. If not the process is repeated.

Once the subset has been successfully correlated using the subset-splitting method there are two sets of WFPs but only one set can be recorded per subset. The set of WFPs that applies to the largest number of pixels in the subset is recorded for the subset since this set of WFPs has more influence on the correlation criterion.

3.2.4 Particle swarm optimisation

Particle swarm optimisation (PSO) is an heuristic optimisation technique which attempts to find the minimum , $\boldsymbol{x}_{opt}$, of an objective function, f such that

$$f : \mathbb{R}^n \to \mathbb{R} \tag{3.48}$$

$$\forall \boldsymbol{x} \in \mathbb{R} : f(\boldsymbol{x}_{opt}) \leq f(\boldsymbol{x}) \tag{3.49}$$

This technique maintains a population of candidate solutions which are referred to as particles. Upon each iteration of the algorithm these particles move through the search space where the movement of the particles is guided by swarm intelligence similar to that of a flock of birds or school of fish. This movement of the particles is termed the velocity of the particles. Letting $\boldsymbol{x}_i(t)$, $\boldsymbol{x}_i(t+1)$ and $\boldsymbol{v}_i(t+1)$ represent the previous position of particle i, the updated position of the particle and the velocity of the particle during the current iteration, the particles movement during an iteration is given as

$$\boldsymbol{x}_i(t+1) = \boldsymbol{x}_i(t) + \boldsymbol{v}_i(t+1). \tag{3.50}$$

This occurs for each particle where each particle has their own velocity vector. The velocity vector of each particle i is calculated as

$$\boldsymbol{v}_i(t+1) = c_3.\boldsymbol{v}_i(t) + c_1.\boldsymbol{r}_1.(\boldsymbol{y}_i(t) - \boldsymbol{x}_i(t)) + c_2.\boldsymbol{r}_2.(\boldsymbol{y}_{best}(t) - \boldsymbol{x}_i(t)) \tag{3.51}$$

where $\boldsymbol{v}_i(t)$ is the previous velocity vector, $\boldsymbol{r}_1$ & $\boldsymbol{r}_2$ are diagonal matrices of random numbers uniformly distributed between 0 and 1, $\boldsymbol{x}_i(t)$ is the previous particle position and c_1, c_2 \$ c_3 are coefficients. Here $\boldsymbol{y}_i(t)$ is the best position that the particle has occupied throughout the iterations defined as

$$\boldsymbol{y}_i(t+1) = \begin{cases} \boldsymbol{y}_i(t) & \text{if} \quad f(\boldsymbol{x}_i(t+1)) \geq f(\boldsymbol{y}_i(t)) \\ \boldsymbol{x}_i(t+1) & \text{if} \quad f(\boldsymbol{x}_i(t+1)) < f(\boldsymbol{y}_i(t)) \end{cases} \tag{3.52}$$

Similarly $\boldsymbol{y}_{best}(t)$ is the best position that any of the particles have occupied throughout the iterations. The first term of Equation 3.51 is the inertia component which tries to keep the particle moving in the same direction it was during the previous iteration. As such c_3 is called the inertia coefficient and its value, in the range $0 \leq c_3 \leq 1.2$ [11], determines whether the particle is accelerated or decelerated the particle in its original direction of motion.

The second term of the equation is the cognitive component which influences the particle to return to regions of the search space where the particle exhibited its best function value. Its coefficient, the cognitive coefficient c_1, falls in the range $0 \leq c_1 \leq 2$. The last term of the equation is the social component which attempts to accelerate the particle towards the region of the search space that contains the best candidate solution found so far. Its accompanying coefficient is the social coefficient, c_2, which has a value within the range $0 \leq c_2 \leq 2$. The cognitive and social components simulate swarm intelligence of the particles.

The technique starts by first initiating the positions and velocities of the particles. These particles are initiated such that they are located in a uniform random distribution throughout the search space. The search space is defined by two vectors which give the lower, $\boldsymbol{b}_l$, and upper bound, $\boldsymbol{b}_u$, for the parameters of the objective function. The velocities of the particles are initiated at random such that they fall within the range $[-\boldsymbol{v}_{range}, \boldsymbol{v}_{range}]$ which is calculated as

$$\boldsymbol{v}_{range} = \boldsymbol{b}_u - \boldsymbol{b}_l. \tag{3.53}$$

After the position and velocities of the particles have been initiated the objective function is evaluated at the initial particle positions. The best position, $\boldsymbol{y}_{best}(t)$, occupied by a particle is then determined from these function values while the best position occupied by each individual particle, $\boldsymbol{y}_i(t)$, is equivalent to its initial position because each particle has only occupied a single position.

At this point the technique begins iterating. First the updated velocity vector of each particle is determined according to Equation 3.51. Within the first iteration the cognitive component of Equation 3.51 cancels out and so does not influence the velocity of the particle. These velocities are applied to the current positions of the particles, which for the first iteration is the initial positions of the particles, according to Equation 3.50. The objective function is evaluated at the updated particle positions and the function values are used to determine the new values for $\boldsymbol{y}_i(t)$ and $\boldsymbol{y}_{best}(t)$.

These iterations are repeated until either the value of $\boldsymbol{y}_{best}(t)$ is smaller than a predefined tolerance or the algorithm exceeds a predefined number of function evaluations. During the iterations it is possible for particles to move outside of the search space. If this occurs the particle is repositioned such that it lies on the boundary of the search space.

particles outside search space initiation of particles and velocities. points as particles which move through the search space avoid gradient so offer different approach to NR

Chapter 4

Synthetic validation

Before the DIC framework presented here can be adopted for Materials Engineering purposes it first needs to be validated. Validation requires proving that the performance of the proposed framework is on par with that of commercial software. The commercial software used for comparative purposes is that of LaVision's Davis software.

this should rather refer to the objectives in the intro

The accuracy of the displacements computed by DIC is dependent on two aspects: correlation of the images and calibration of the camera system. Correlation deals with computing the displacement fields between images in units of pixels. As such correlation operates solely within the sensor coordinate system. Calibration first determines the parameters of the pinhole camera model for the camera system used and then uses these parameters to relate the displacements in the sensor coordinate system, determined by correlation, to displacements in the world coordinate system.

Thus the correlation component of the proposed framework can be validated in a standalone manner. However calibration cannot be validated separately from correlation because it operates on the displacements determined from correlation. Therefore validation has been broken up into two parts.

First the correlation framework is validated in this chapter and then the overall DIC framework is validated in the chapter to follow. The proposed correlation framework is validated by showing that the precision and accuracy of its displacement results are on par with that of the Davis software.

4.1 Precision and accuracy

For a measurement method accuracy refers to how close a measured value is to the true value while precision refers to how reproducible this accuracy is. Precision and accuracy are quantified by the systematic and random errors respectively. For a measurement method the error is defined as the difference between the true value and the measured value. In terms of correlation if the true displacement of subset i in the x and y directions are given as u_i^{true} and v_i^{true} and the measured values are u_i^{meas} and v_i^{meas} then the error in the x and y directions, Δu_i^e and Δv_i^e, are given as

$$\Delta u_i^e = u_i^{meas} - u_i^{true} \quad \text{and} \quad \Delta v_i^e = v_i^{meas} - v_i^{true}. \tag{4.1}$$

A pair of such errors is associated with the measured displacement of each subset. Therefore if n many subsets are used during correlation then there are n many error pairs associated with the calculated displacement field. The systematic and random errors are statistical measurements based on these groups of errors.

The systematic error is quantified by the mean absolute error in both directions as

$$\Delta u^{mae} = \frac{\sum_i^n |\Delta u_i^e|}{n} \quad \text{and} \quad \Delta v^{mae} = \frac{\sum_i^n |\Delta v_i^e|}{n}. \tag{4.2}$$

The random error is quantified by the standard deviation of the absolute error in both directions given as

$$\sigma_u = \sqrt{\frac{\sum_i^n \left[|\Delta u_i^e| - \Delta u^{mae}\right]^2}{n-1}} \quad \text{and} \quad \sigma_v = \sqrt{\frac{\sum_i^n \left[|\Delta v_i^e| - \Delta v^{mae}\right]^2}{n-1}}. \tag{4.3}$$

The absolute of the error is used in Equations 4.2 and 4.3 so that positive and negative errors do not cancel each other out which would skew the results. Additionally, according to Amiot et al. [4], to ensure statistical independence in the systematic and random errors it is important that the subsets evaluated do not overlap.

et al. correct?

4.2 Synthetic images

It is clear that the true displacement of each subset needs to be known in order to evaluate Equations 4.1, 4.2 and 4.3. This requires knowing the true displacement field that exists between two images which is not possible for images captured by a camera. Instead synthetic images need to be generated for this purpose.

A synthetic image is an image in which the light intensities of the pixels are calculated according to an algorithm instead of using a CCD array to capture the intensities of actual light rays. The algorithm used in this project makes use of a speckle function and a displacement field to create an image set. The speckle function, f_{sp}, is a two-dimensional, continuous function which defines the light intensity at any location in the reference image. Consequently the matrix of the reference image, $\boldsymbol{F}^{syn}$, is generated by sampling the speckle function at pixel centre locations. More specifically the matrix element of row

i and column j of the reference image, F_{ij}^{syn}, is calculated by sampling the speckle function at pixel centre location (x_j^{pix}, y_i^{pix}) as

$$F_{ij}^{syn} = f_{sp}\left(x_j^{pix}, y_i^{pix}\right) \tag{4.4}$$

The displacement field consists of two continuous, two-dimensional functions which define the displacement in the x and y directions of the speckle function throughout the image. These are f_u and f_v respectively. Although these functions describe how the speckle pattern should deform they do not have to be applied to the speckle function directly to achieve this. Instead the inverse of the displacement field (displacement in the direction opposite to what these functions describe) can be applied to the pixel centre locations. In doing so the locations where the speckle function is sampled change thereby simulating the deformation between the images that is described by the displacement field. As such the matrix element of row i and column j of the deformed image, G_{ij}^{syn}, is calculated by sampling the speckle function as

$$G_{ij}^{syn} = f_{sp}\left(x_j^{disp}, y_i^{disp}\right) \tag{4.5}$$

$$\text{where} \quad x_j^{disp} = x_j^{pix} - f_u(x_j^{pix}, y_i^{pix}) \quad \text{and} \quad y_i^{disp} = y_i^{pix} - f_v(x_j^{pix}, y_i^{pix}). \tag{4.6}$$

Therefore the true displacement experienced by a subset can be determined by sampling the displacement field functions at the location of the subset centre.

There are additional advantages to using synthetic images to validate the proposed correlation framework. Firstly the synthetic images generated contain minimal noise. This allows the performance of the correlation framework to be evaluated for optimal conditions with respect to noise or noise of known severity can be applied to the images in order to investigate the limitations of the framework with respect to noise.

Secondly control of the displacement field allows the capabilities of the correlation framework to be evaluated for displacement fields of varying complexity. More specifically, image sets of continuous, low-complexity displacement fields can be used to determine the best performance the framework can achieve. Conversely images of discontinuous, complex displacement fields will give an indication of the limitations of the framework.

4.2.1 Generating synthetic images

This section gives a detailed breakdown of the image generation process which is based on work done by Orteu et al. [21]. A flow diagram of the process is presented in Figure 4.1 which uses lines numbers to indicate which lines of code in Appendix A perform the various tasks of the process. The core operations of this flow diagram are then explained in the sections that follow.

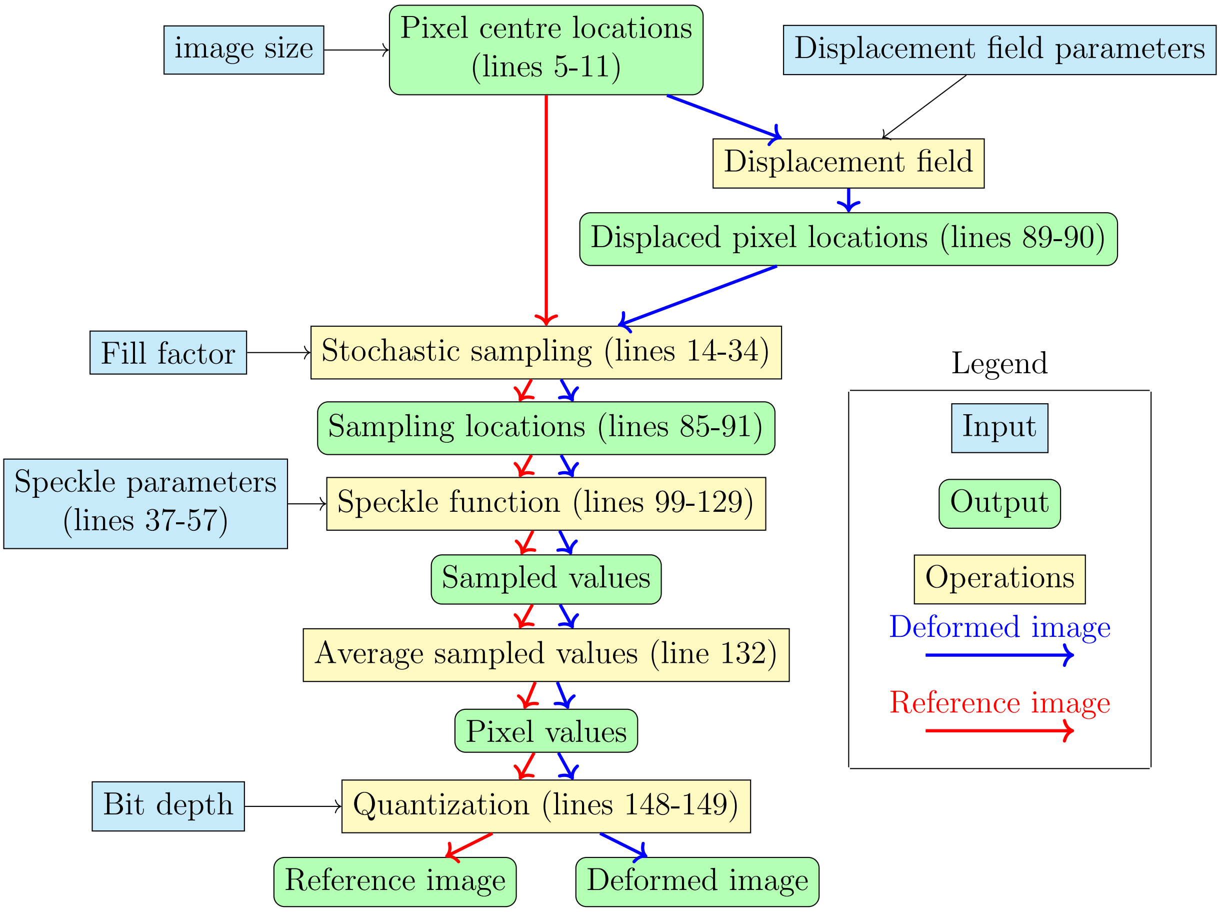


image size
Pixel centre locations (lines 5-11)
Displacement field parameters
Displacement field
Displaced pixel locations (lines 89-90)
Fill factor
Stochastic sampling (lines 14-34)
Sampling locations (lines 85-91)
Speckle parameters (lines 37-57)
Speckle function (lines 99-129)
Sampled values
Average sampled values (line 132)
Pixel values
Bit depth
Quantization (lines 148-149)
Reference image
Deformed image
Legend
Input
Output
Operations
Deformed image
Reference image

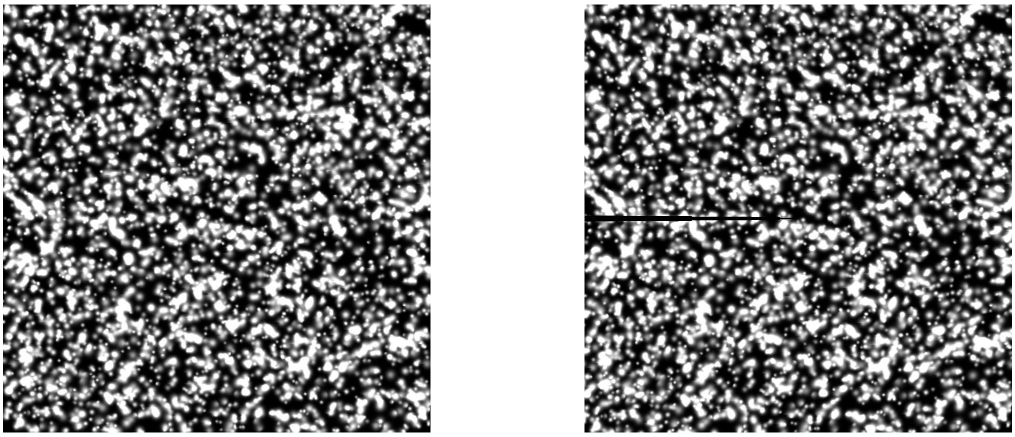
Figure 4.1: Flow diagram for the process of generating synthetic images

reogranise the flow diagram so inputs on left, straight arrows

First the image size is defined by the number of pixels in the x and y directions. This is used to determine a grid of points which correspond to pixel centre locations of the reference image.

Stochastic sampling is then performed on these pixel centre locations such that a cluster of sampling locations is created for each pixel centre location. The speckle function is then created by assigning values to the parameters of each speckle in a predominantly random manner. The speckle function is then evaluated at the sampling locations for the reference image.

Thereafter for each pixel centre location the group of sampled values, that correspond to its cluster of sampling locations, is averaged in order to obtain a single light intensity for each pixel. Finally the light intensities of the pixels are quantified in order to simulate the analogue to digital conversion of the CCD array. This is done using 12 bit quantization which is consistent with the cameras used during the experimental tests. This results in the final reference image.



(a) Reference image

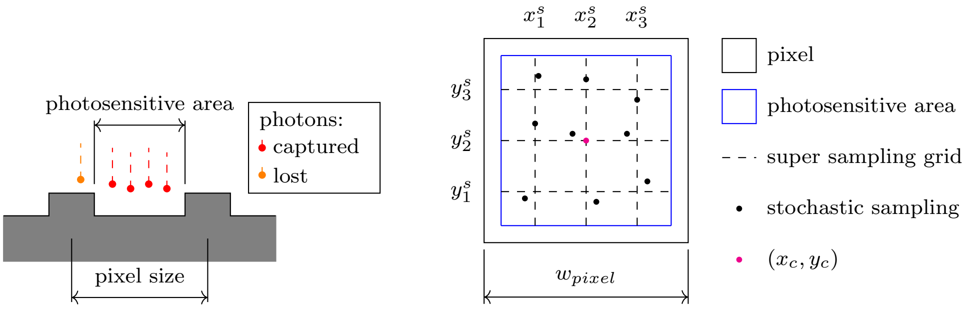
(b) Deformed image

Figure 4.2: Example of a reference and deformed synthetic image

Generating the deformed image involves a minor adjustment to the process used for generating the reference image. For the deformed image the pixel centre locations, determined from the image size, are displaced according to Equation 4.6 in order to obtain modified pixel centre locations. Stochastic sampling is then performed on these modified pixel centre locations in order to obtain the sampling locations for the deformed image. Thereafter the process is the same as for the reference image except that the speckle function does not need to be created again; the same speckle function is used for the deformed image. An example of the reference and deformed image is given in Figure 4.2.

Stochastic sampling

Sampling the speckle function at pixel centre locations is not fully representative of how digital images are actually captured. The light intensity for each pixel in a captured image is determined by the voltage generated by the pixel's corresponding CCD array element. However an element of the CCD array outputs a voltage that corresponds to all the light that falls upon its photosensitive surface and not just the light that falls at its centre as shown in Figure 4.3a.



(a) Fill factor (b) Stochastic sampling

Figure 4.3: Illustration of the effect of the fill factor and the stochastic sampling technique

Furthermore, as seen in the figure, the photosensitive surface of a CCD array element is not the same size as the pixel it represents. This is because of the space required for the electrical components, such as electrodes and transistors, which connect the elements of the CCD array to a circuit which measures their voltages. The ratio of the photosensitive area to the total area of an element of the CCD array is referred to as the fill-factor, ω.

Therefore to accurately simulate the process of the CCD array a method called stochastic sampling is used which is a form of super sampling. Super sampling, in this context, refers to sampling the speckle function multiple times, at different positions that fall within the photosensitive area of the same pixel (CCD array element), and averaging these function values to obtain the light intensity value for that pixel.

ref here too?

The simplest method of doing this is to create a grid of points that fall within the photosensitive area of a pixel and evaluate the speckle function at these points as shown in Figure 4.3b. Thus if a pixel of width and height w_{pixel} has its centre located at point (x_c, y_c) then the location of the super sampling grid points in the x and y directions are given as x_j^s and y_i^s.

$$x_j^s = x_c - \frac{w_{pixel}}{2}\sqrt{\omega}\left(1 + \frac{2j-1}{n}\right) \quad \text{where} \quad j = 1, ..., n \tag{4.7}$$

$$y_i^s = y_c - \frac{w_{pixel}}{2}\sqrt{\omega}\left(1 + \frac{2i-1}{n}\right) \quad \text{where} \quad i = 1, ..., n. \tag{4.8}$$

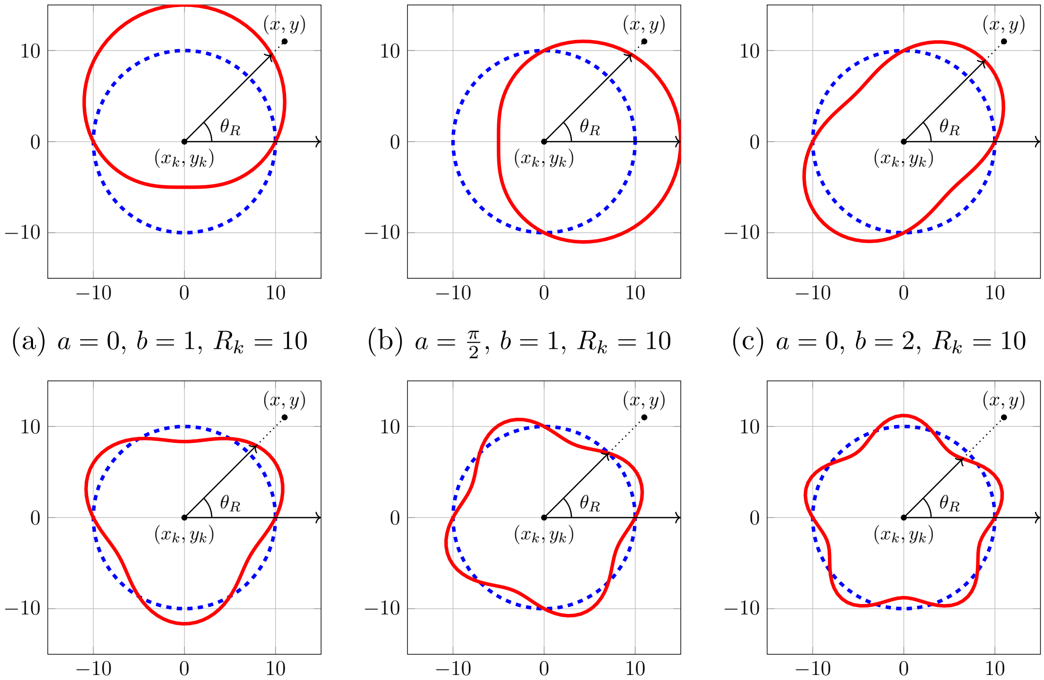
Here n^2 is the number of sampling points contained within the photosensitive area of a pixel. Super sampling in this manner however can lead to aliasing in the image which is not representative of the CCD array process. To avoid this the grid can be jittered whereby each grid point is translated by a random amount, less than the grid spacing, in a random direction. The speckle function is then sampled at these jittered positions shown in Figure 4.3b. This is referred to as stochastic sampling. By doing this the aliasing effect is reduced at the expense of slightly increasing the noise contained in the image [9].

Speckle function

The function used to create a speckle pattern is based on that proposed by B. Pan et al. [10]. However, because this proposed speckle function creates speckles of constant radius, the resulting speckle pattern contains only circular speckles which is not representative of real speckle patterns created by spaying paint.

In order to account for this the radius is modified to be a function of the angle between the radius line and the x-axis, referred to as the radius angle (θ_R in Figure 4.4). In other words the radius is a function of the arctan of the vector suspended between the speckle centre, (x_k, y_k), and the investigated point in the image, (x, y).

end with comma'ed variable correct?



(a) $a = 0$, $b = 1$, $R_k = 10$ (b) $a = \frac{\pi}{2}$, $b = 1$, $R_k = 10$ (c) $a = 0$, $b = 2$, $R_k = 10$

(d) $a = 0$, $b = 3$, $R_k = 10$ (e) $a = 0$, $b = 4$, $R_k = 10$ (f) $a = 0$, $b = 5$, $R_k = 10$

Figure 4.4: The affect that the frequency, b_k, and phase shift, a_k, of the sinusoid have on the shape and orientation of the resulting speckle. The units of the axes are in pixels, the red line indicates the modified radius according to equation 4.9 and the blue line indicates the base radius R_k.

The speckle function used in this project is given as

$$f_{sp}(x,y) = \sum_{k=1}^{n} I_k \exp\left[-\frac{(x-x_k)^2 + (y-y_k)^2}{\left(\frac{R_k}{2}.f_{amp}(b_k).\sin\left(\left(\arctan\left(\frac{y-y_k}{x-x_k}\right)+a_k\right).b_k\right)+R_k\right)^2}\right] \tag{4.9}$$

$$\text{where} \quad f_{amp}(b_k) = 0.1 + 2.44.e^{-b_k} \tag{4.10}$$

$$\{a_k \in \mathbb{R} : 0 \leq a_k \leq 2\pi\} \tag{4.11}$$

$$\{b_k \in \mathbb{Z} : 1 \leq b_k \leq 5\}. \tag{4.12}$$

Here $\arctan\left(\frac{y-y_k}{x-x_k}\right)$ is the radius angle (θ_R), I_k is the light intensity and R_k is the base radius of speckle k. This function, f_{sp}, determines the light intensity at a sample point, (x, y), in the image by summing the light contribution of all n speckles to this point. The light contribution of a single speckle to a sample point decays exponentially based on the value of the fraction contained with the square brackets of Equation 4.9.

a and b notation needs to be checked

The numerator of this fraction gives the squared distance between the sample point and the speckle centre while the denominator is the expression for the radius of the speckle, in the direction of the sample point, squared. This expression gives the radius as a sinusoid which is a function of the radius angle, has a maximum amplitude of half the base radius and has a baseline equal to the base radius. Therefore the radius varies as the radius angle changes which allows the speckle to have an irregular shape.

midline or vertically shifter by base radius

The actual shape of the speckle is determined by the variable b_k which alters the frequency of the sinusoid. The different speckle shapes possible are illustrated in Figure 4.4. Additionally the variable a_k applies a phase shift to the sinusoid which has the effect of rotating the speckle shape about the speckle centre. This is clear when comparing Figures 4.4a and 4.4b.

can i say this instead of putting it in the figure?

Therefore randomly assigning values to a_k and b_k for each speckle, from the sets of numbers defined in Equations 4.11 and 4.12, enforces a large degree of variance in the speckle shapes contained within the final speckle pattern. The reason for using a continuous function to define the speckle pattern is so that no interpolation is required when sampling the speckle pattern since this interpolation would introduce undesired bias in the generated images.

The purpose of the function f_{amp} in Equation 4.10 is to reduce the amplitude of the sinusoid as the frequency increases. If this function is excluded from Equation 4.9 the resulting speckle shape, for larger values of b_k, exhibits a star shape which is an unrealistic speckle shape. The equation of this function was found through trial and error.

4.2.2 Displacement fields

A variety of displacement fields have been used to create synthetic image sets. The first two give theoretical displacement fields for a plate of material in the vicinity of a stress concentration which causes complex displacement fields. The final displacement field is designed to test the limitations of the framework so that the point at which the proposed framework breaks down can be compared to that of commercial software. (NOTE: still need to design this displacement field and present subsequent discussion of it.)

The displacement field equations are given as a function of the sampling point, (x, y), and an additional parameter which controls the severity of the displacement field. The reason for this is so that an image set containing multiple deformed images, of increasing deformation, can be created by altering this variable.

Plate with hole

The displacement field experienced by an infinite plate of material containing a hole subjected to stress S, shown in Figure 4.5, has been calculated in Appendix B.1. The final form of these displacement fields is given as

$$\begin{aligned} f_u(x,y,S) = \frac{S\,a}{8\,\mu}\Bigg[& \frac{2\,a}{\sqrt{x^2+y^2}}\left(\cos\left(3\,\mathrm{atan2}(y,x)\right) + \cos\left(\mathrm{atan2}(y,x)\right)\,(k+1)\right) \\ & - \frac{2\,a^3}{\left(\sqrt{x^2+y^2}\right)^3}\cos\left(3\,\mathrm{atan2}(y,x)\right) + \frac{\sqrt{x^2+y^2}}{a}\cos\left(\mathrm{atan2}(y,x)\right)\,(k+1)\Bigg] \end{aligned} \tag{4.13}$$

$$\begin{aligned} f_v(x,y,S) = \frac{S\,a}{8\,\mu}\Bigg[& \frac{2\,a}{\sqrt{x^2+y^2}}\left(\sin\left(3\,\mathrm{atan2}(y,x)\right) - \sin\left(\mathrm{atan2}(y,x)\right)\,(k-1)\right) \\ & - \frac{2\,a^3}{\left(\sqrt{x^2+y^2}\right)^3}\sin\left(3\,\mathrm{atan2}(y,x)\right) + \frac{\sqrt{x^2+y^2}}{a}\sin\left(\mathrm{atan2}(y,x)\right)\,(k-3)\Bigg] \end{aligned} \tag{4.14}$$

Note that atan2 is a Matlab function which computes the arctan of a vector and returns the angle corresponding to the correct quadrant within which that vector lies. These displacement fields are only valid for x and y positions that lie outside of the radius of the hole. Here μ is the shear modulus and k is Kolosov's constant. The values for the shear modulus and Kolosov's constant are determined from the elastic modulus, E, and Poisson's ratio, v, under the assumption of plane stress as

$$\mu = \frac{E}{2(1+v)} \quad \text{and} \quad k = \frac{3-v}{1+v}. \tag{4.15}$$

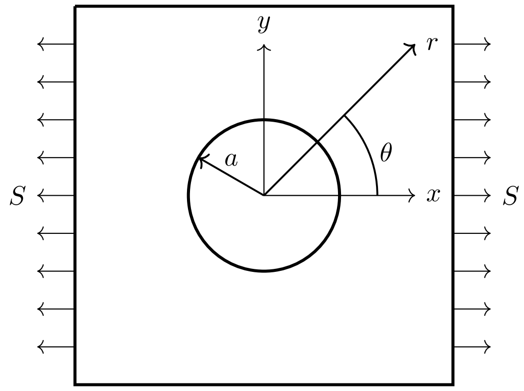
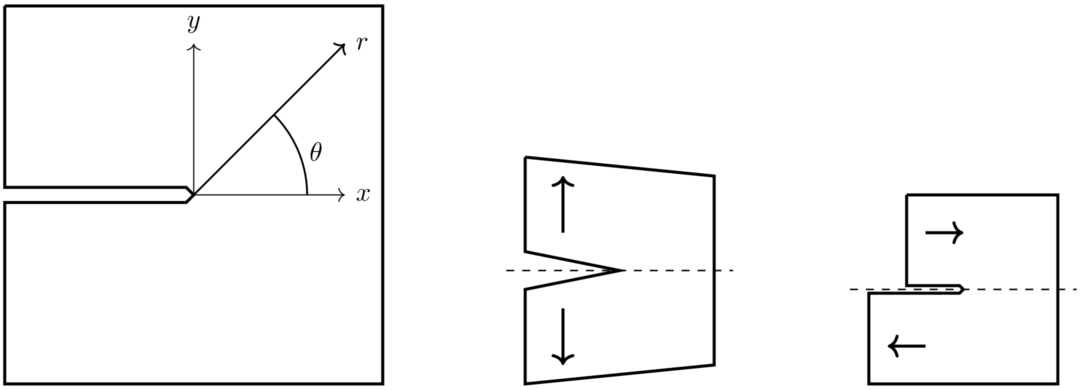


Figure 4.5: Plate with hole

Plate with crack

The second stress concentration considered is that of a crack in a large plate illustrated in Figure 4.6a. The plate can deform in multiple modes which depend on the boundary conditions it experiences. The two modes considered are that of mode I and II which are illustrated in Figures 4.6b and 4.6c. For mode I the material above and below the crack move in opposite directions perpendicular to the crack surface such that the crack opens up. Mode II differs in that the material above and below the crack move in opposite directions parallel to the crack surface such that they slide past one another.



(a) Crack in a large plate

(b) Mode I

(c) Mode II

Equations that describe the displacement of the specimen around the crack, for both modes, can be derived using the Williams approach [31]. These equations, which were converted to Cartesian coordinates in Appendix B.2, are given as

Mode I:

$$f_u(x,y,\boldsymbol{a}) = \sum_{n=1}^{\infty} \frac{(x^2+y^2)^{\frac{n}{4}}}{2\mu} a_n \Bigg[\left(k + \frac{n}{2} + (-1)^n\right) \cos\left(\frac{n.\text{atan2}(y,x)}{2}\right) - \frac{n}{2} \cos\left(\frac{(n-4)\text{atan2}(y,x)}{2}\right) \Bigg] \tag{4.16}$$

$$f_v(x,y,\boldsymbol{a}) = \sum_{n=1}^{\infty} \frac{(x^2+y^2)^{\frac{n}{4}}}{2\mu} a_n \Bigg[\left(k - \frac{n}{2} - (-1)^n\right) \sin\left(\frac{n.\text{atan2}(y,x)}{2}\right) + \frac{n}{2} \sin\left(\frac{(n-4)\text{atan2}(y,x)}{2}\right) \Bigg] \tag{4.17}$$

Mode II:

$$f_u(x,y,\boldsymbol{b}) = -\sum_{n=1}^{\infty} \frac{(x^2+y^2)^{\frac{n}{4}}}{2\mu} b_n \Bigg[\left(k + \frac{n}{2} - (-1)^n\right) \sin\left(\frac{n.\text{atan2}(y,x)}{2}\right) - \frac{n}{2} \cos\left(\frac{(n-4)\text{atan2}(y,x)}{2}\right) \Bigg] \tag{4.18}$$

$$f_v(x,y,\boldsymbol{b}) = \sum_{n=1}^{\infty} \frac{(x^2+y^2)^{\frac{n}{4}}}{2\mu} b_n \Bigg[\left(k - \frac{n}{2} + (-1)^n\right) \cos\left(\frac{n.\text{atan2}(y,x)}{2}\right) + \frac{n}{2} \cos\left(\frac{(n-4)\text{atan2}(y,x)}{2}\right) \Bigg] \tag{4.19}$$

$$\text{where} \quad \boldsymbol{a} = [a_1, a_2, .., a_n] \quad \text{and} \quad \boldsymbol{b} = [b_1, b_2, .., b_n]. \tag{4.20}$$

Here a_n and b_n are coefficients which control the severity of the corresponding term (in square brackets). As in Section 4.2.2 μ is the shear modulus and k is Kolosov's constant determined according to Equation 4.15. Although the sum in the above equations is stated as infinite, a finite number of terms will approximate the displacement field sufficiently.

state how many summations i used

mixed mode

Chapter 5

Experimental validation

As stated in Chapter 4 the calibration part of the DIC framework cannot be validated in a standalone manner. Instead the validity of the overall DIC framework is investigated in this chapter using experimental image sets captured of specimens as a load is applied to them. As such performing DIC on these image sets requires not only determining the displacement fields between images but also relating these displacement fields, which are in the sensor coordinate system, to the world coordinate system.

Therefore the quality of the resulting displacements are dependent on the performance of three processes that make up the DIC framework. Namely the correlation process, the process of determining the calibration parameters and how these calibration parameters are used to convert from the sensor coordinate system to the world coordinate system.

DIC is performed on these image sets using both the proposed DIC framework and LaVision's DaVis software (version) . The resulting displacements in world coordinates are then compared to give an indication of how well the proposed framework performs relative to a commercial software package.

give version

The actual displacement fields for these images are not known and so the results need to be analysed in a qualitative manner. Instead of determining the precision and accuracy of the results the results will be compared as a normalised difference between the two

applying speckle pattern (how was done?)

5.1 Experimental plan

This section outlines the equipment, equipment settings and specimens used in order to capture image sets for the purpose of experimental validation.

5.1.1 Equipment

The first piece of equipment that was used was the MTS Criterion Model 44 tensile testing machine which was used to apply tensile loading forces to the specimens. The rate of cross-head displacement was controlled on this machine to ensure that the specimens remained in quasi-static equilibrium throughout the tests. The second piece of equipment was that of LaVision's DIC camera system which was used to capture images of the specimens as they were loaded. This camera system consists of a 5 megapixel CCD camera (Imager E-lite 5M), two light sources and a programmable timing unit (PTU). The PTU controls the camera and light sources such that images are captured at a specific rate with the camera and light sources being triggered simultaneously.

Lastly a calibration plate was used in order to obtain calibration images such that the DIC software can determine the calibration parameters needed to relate displacements in sensor coordinates to world coordinates. The SPECIFIC NAME calibration plate was used.

grips?

5.1.2 Specimens

Two types of specimens where chosen for experimental validation. These were a tension specimen with a hole in it and the modified Arcan specimen.

Speckle patterns were applied to the specimens over the region of interest prior to testing. This was done using cans of spray paint. First a base layer of white paint was applied by spraying the can at the recommended distance from the specimens. Once this dried black speckles were created on top of this base layer of white by spraying the can of paint from further away.

illustrate the speckle patterns?

Tension specimen with hole

This specimen is based on a standard tension specimen designed according to ASTM E8 standard [15] which has been modified to have a 8 mm diameter hole cut in the middle. The hole acts as a stress concentration leading to more complex displacement fields occurring around it. The specimen was made from 6 mm thick aluminium sheets so that significant deformation could be achieved while not exceeding the limitations of the tensile testing machine.

loading rate

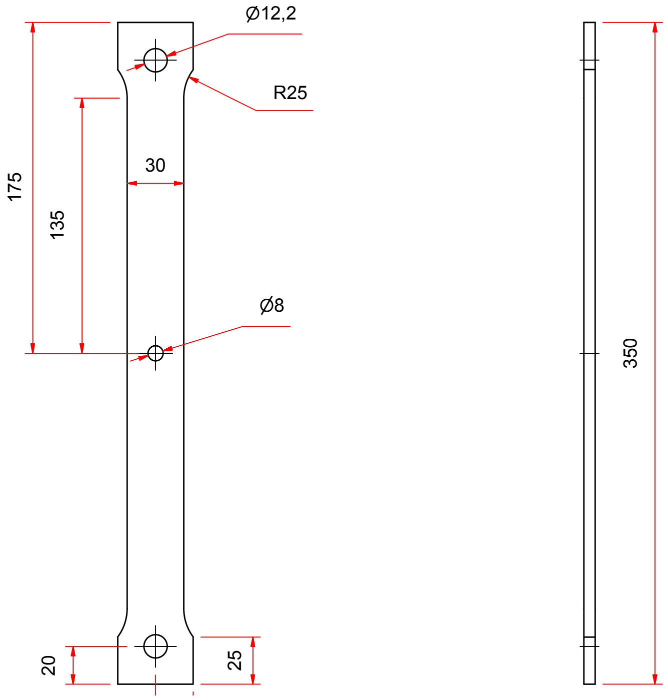


Figure 5.1: Tension specimen with hole (I will redo this image such that the specimen lies horizontally so that it takes up less space)

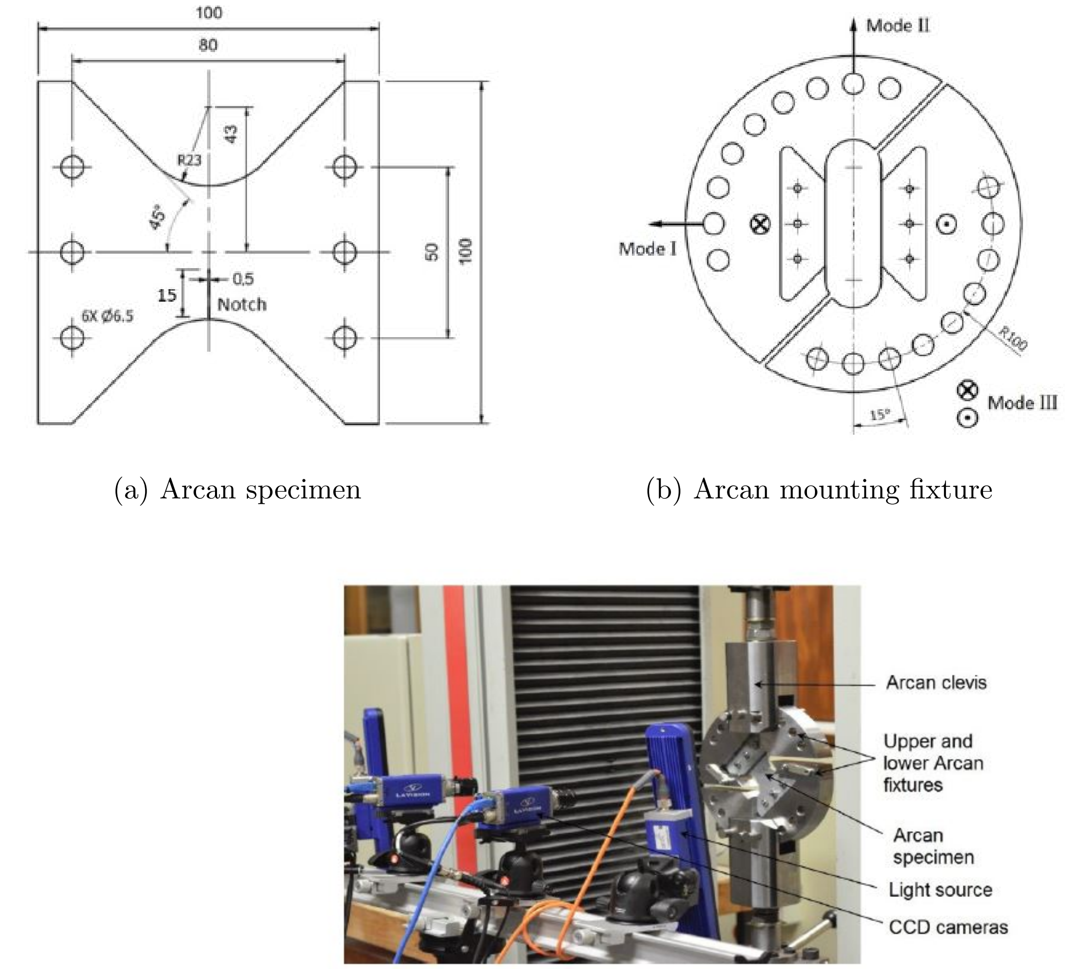
Modified Arcan specimen

There are many specimen geometries that are designed to investigate the propagation of a crack in the material such as the compact tension specimen, the single-edge-notched bend specimen and the arc-shaped tension specimen [5]. However the modified Arcan specimen is unique in that it can investigate pure mode I, pure mode II or mixed mode I-II crack propagation.

The modified Arcan specimen, shown in Figure 5.2a, and its mounting fixture, shown in Figure 5.2b, were designed according to the guidelines of Banks-Sills and Arcan [7] [1]. These Arcan specimens were laser cut from a 6 mm thick sheet of polymethyl methacrylate (PMMA). PMMA was used because it exhibits linear-elastic brittle fracture so that localised deformation in the vicinity of the crack tip can be minimised. A sharp pre-crack was created in each specimen from the tip of the starter notch, located 15mm from the filleted edge of the specimen, until 20mm from the filleted edge.

si units for length

[1] These specimens where designed, manufactured and tested by Matthew Molteno and Johan



(a) Arcan specimen

(b) Arcan mounting fixture

(c) Arcan test set up

Figure 5.2: Arcan components and test setup (these are Matt's images - I will make my own)

The specimen is loaded in the tensile testing machines as shown in Figure 5.2c where the specimen is bolted to the two fixture halves and each fixture half is mounted to a clevis using pins through its outer holes. Each fixture half is mounted to a clevis using three pins through its outer holes such that lateral and rotational movement between the two is constrained. The mode of crack propagation is dependent on the angle of the plane of the pre-crack relative to the line of loading of the clevises. Therefore by changing the which holes of the fixture halves are pinned to the clevises this angle can be controlled. The holes for mode I and mode II crack propagation are indicated in Figure 5.2b.

The fixture was loaded using an MTS tensile tester with a cross-head speed of 0.25 mm/min. Images were captured at a rate of 1Hz using the LaVision DIC camera system (5MP CCd cameras).

The reason for including a This specimen was chosen so that the subset splitting functionality implemented within the proposed DIC framework can be tested for cracks at different angles . As such

lines of discontinuity at different angles

.

.

.

.

how rotate specimen to change mode of loading designd according to guidelines origitnal specimen and experiments done by pictures of them. why controlling mode is important

5.1.3 Speckle patterns

Speckle patterns were applied to the specimens prior to testing. First a base layer of white paint was applied to the specimens over the region of interest of the specimen. Thereafter black speckles were created ontop of this white layer by using a spray can

Chapter 6

Results

This section provides the results of the Matlab codes in order to prove that it works. The results will be presented here in accordance to the layout of the project sections.

6.1 Part A

Part A dealt with calibration only. The intrinsic parameters determined were

$$\boldsymbol{K} = \begin{bmatrix} 657.62840 & 0.30335 & 303.72868 & 0 \\ 0 & 658.47823 & 245.69177 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{6.1}$$

The radial distortion parameters were determined to be $k_1 = -0.24885$ and $k_2 = 0.11014$. Using these calibration parameters the image "img01.tiff" was undistorted and is provided in figure 6.2. Comparing this to the reference image in figure 6.1 it is clear that the calibration parameters determined are sufficiently accurate however differences between the images are noticeable. These differences are likely as a result of additional distortion types that have not been taken into account due to the complexity that these additional distortions would introduce.

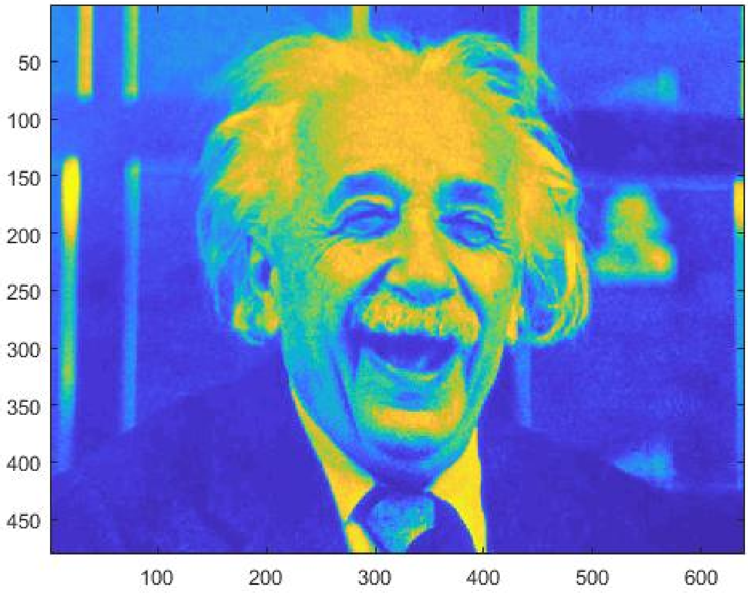


Figure 6.1: Reference image for comparison purposes

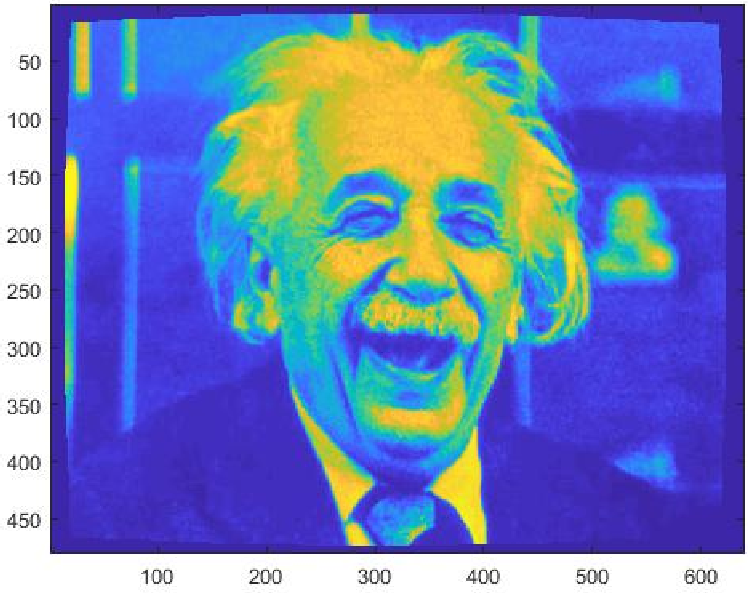


Figure 6.2: The distorted image ”img01.tiff” which has been corrected for distortions

6.2 Part B

Part B deals with the correlation process only. The warp function parameters determined by the Matlab code are presented below.

$$u = 5.03754 \tag{6.2}$$

$$\frac{\partial u}{\partial x} = 0.04073 \tag{6.3}$$

$$\frac{\partial u}{\partial y} = -0.02176 \tag{6.4}$$

$$v = -1.03562 \tag{6.5}$$

$$\frac{\partial v}{\partial x} = -0.00915 \tag{6.6}$$

$$\frac{\partial v}{\partial y} = 0.03048 \tag{6.7}$$

The corresponding correlation coefficients are

$$C_{ZNSSD} = 0.00587 \tag{6.8}$$

$$C_{ZNCC} = 0.99706. \tag{6.9}$$

Both correlation coefficients have been presented here in order to illustrate that the C_{ZNCC} coefficient is much easier to interpret. The determined warp function parameters were used to undeform the image ”img02.tiff” to obtain figure 6.4. Comparing this to the reference figure in figure 6.3 it can be seen that the determined warp function parameters are sufficiently accurate.

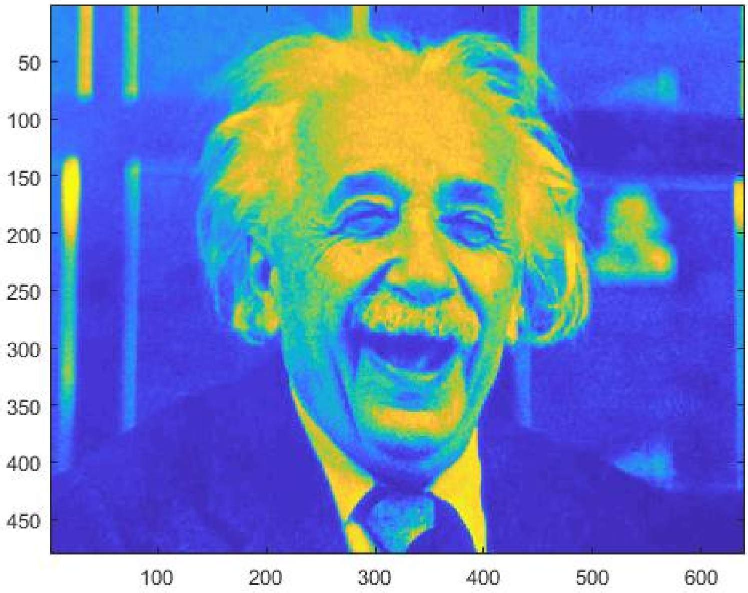


Figure 6.3: Reference image for comparison purposes

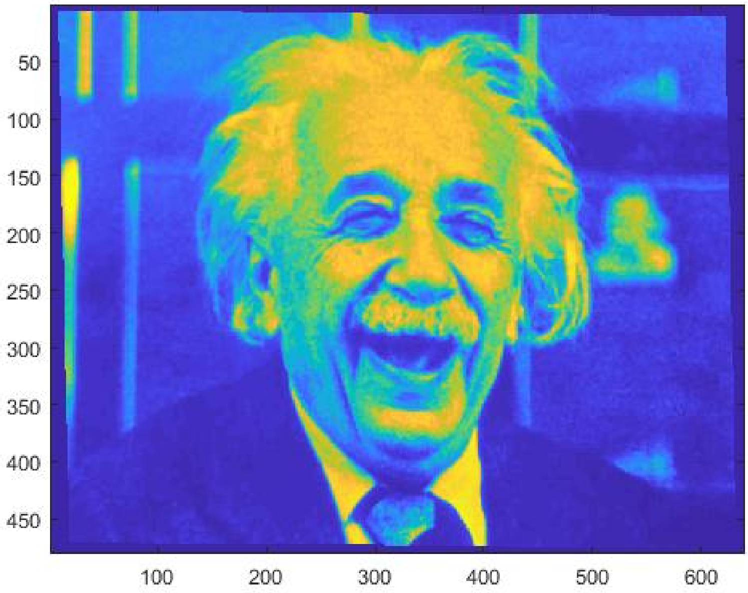


Figure 6.4: The distorted image "img02.tiff" which has been corrected for distortions

6.3 Part C

Part C combine both Part A and Part B. The calibration code of Part A is used to determine the calibration parameters for the camera system. These camera parameters are then used to undistort image "img03.tif" such that it is free of lens distortion. Then the correlation code is applied to this undistorted image in order to determine the warp function parameters. These warp function parameters are then used to undeform the image. This undeformed image is shown in figure 6.6. Comparing this image to the original undistorted image in figure 6.5 it seems that the calibration parameters and warp function parameters were sufficiently accurate. However the actual warp function parameters differ from those of Part B as shown below and the correlation coefficient is not as good as in Part B.

$$u = 5.11364 \tag{6.10}$$

$$\frac{\partial u}{\partial x} = 0.04348 \tag{6.11}$$

$$\frac{\partial u}{\partial y} = -0.02214 \tag{6.12}$$

$$v = -1.06509 \tag{6.13}$$

$$\frac{\partial v}{\partial x} = -0.00918 \tag{6.14}$$

$$\frac{\partial v}{\partial y} = 0.03071 \tag{6.15}$$

$$C_{ZNSSD} = 0.00838 \tag{6.16}$$

$$C_{ZNCC} = 0.99581 \tag{6.17}$$

The reason that these warp function parameters differ from those of Part B is that the this method of using the calibration parameters to undistort the image prior to performing correlation on the image is not correct. This is because interpolation is used to undistort the image and additional interpolation is used within the correlation code.

Thus by introducing an extra case of interpolation the reliability of the data is compromised. This is reflected in the correlation coefficients which differ from those in Part B. Note that the standard deviations of the warp function parameters for both Part B and Part C were on the order of magnitude of 10^{-13} sampled over one thousand correlation runs. The calibration parameters for Part C are not presented since they are the same as the values in Part A.

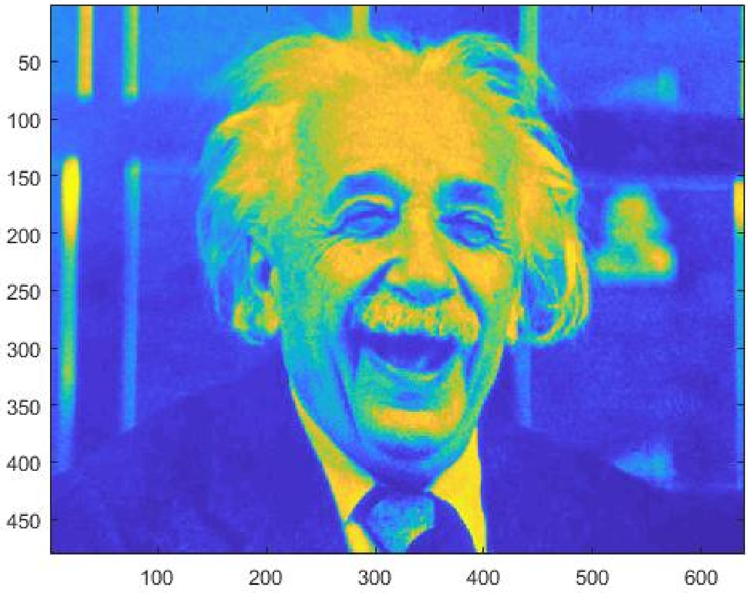


Figure 6.5: Reference image for comparison purposes

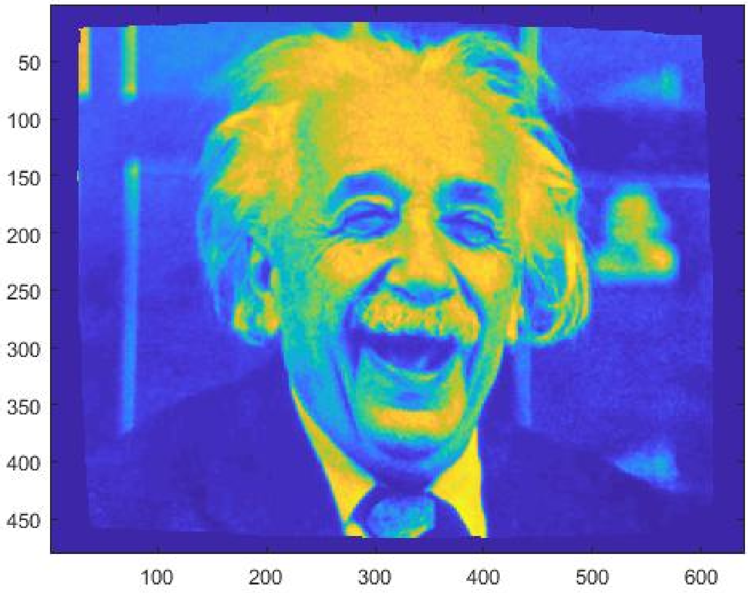


Figure 6.6: The distorted image ”img03.tiff” which has been corrected for distortions

Chapter 7

Conclusion

This document outlines the Digital Image Correlation process. It presents the background information on camera optics and the mathematical relation used to model these optics in a camera model. Thereafter the calibration process is discussed which determines the parameter values for the camera model. Correlation is then reviewed with focus on the inverse compositional Lucas-Kanade algorithm. Finally the results of the Matlab codes are given in order to illustrate that the methods presented here are correct.

Bibliography

[1] ASTM E8 / E8M 16a. Standard test methods for tension testing of metallic materials. Technical report, ASTM, West Conshohocken, PA, 2016.

[2] ASTM E399 17. Standard test method for linear-elastic plane-strain fracture toughness kic of metallic materials. Technical report, ASTM, West Conshohocken, PA, 2017.

[3] YI Abdel-Aziz. Direct linear transformation from comparator coordinates into object space in close-range photogrammetry. In *Proceedings of the ASP Symposium on Close-Range Photogrammetry, 1971*, pages 1–18. American Society of Photogrammetry, 1971.

[4] Fabien Amiot, Michel Bornert, Pascal Doumalin, J-C Dupré, Marina Fazzini, J-J Orteu, Christophe Poilâne, Laurent Robert, René Rotinat, Evelyne Toussaint, et al. Assessment of digital image correlation measurement accuracy in the ultimate error regime: main results of a collaborative benchmark. *Strain*, 49(6):483–496, 2013.

[5] ASTM. Standard test method for linear-elastic plane strain fracture toughness kic of metallic materials. *E 399-12*, 2012.

[6] S. Baker and I. Matthews. *Lucas-Kanade 20 Years on: A Unifying Framework*. Number pt. 1 in Technical report. Carnegie Mellon University, The Robotics Institute, 2002.

[7] Leslie Banks-Sills and Mircea Arcan. A compact mode ii fracture specimen. In *Fracture Mechanics: Seventeenth Volume*. ASTM International, 1986.

[8] JR Barber. Solid mechanics and its applications, elasticity, 2002.

[9] Kristof Beets and Dave Barron. Super-sampling anti-aliasing analyzed. 2000.

[10] Pan Bing, Xie Hui-Min, Xu Bo-Qin, and Dai Fu-Long. Performance of sub-pixel registration algorithms in digital image correlation. *Measurement Science and Technology*, 17(6):1615, 2006.

[11] James Blondin. Particle swarm optimization: A tutorial. *Availaible from: http://cs. armstrong. edu/saad/csci8100/pso tutorial. pdf*, 2009.

[12] Wilhelm Burger. Zhangs camera calibration algorithm: In-depth tutorial and implementation. Technical Report HGB16-05, University of Applied Sciences Upper Austria, School of Informatics, Communications and Media, Dept. of Digital Media, Hagenberg, Austria, May 2016.

[13] Luc Chevalier, Sylvain Calloch, François Hild, and Yann Marco. Digital image correlation used to analyze the multiaxial behavior of rubber-like materials. *European Journal of Mechanics-A/Solids*, 20(2):169–187, 2001.

[14] Fu-Pen Chiang. Micro-/nano-speckle method with applications to materials, tissue engineering and heart mechanics. *Strain*, 44(1):27–39, 2008.

[15] ASTM American Society for Testing and Materials. *Standard test methods for tension testing of metallic materials.* ASTM international, 2009.

[16] Amos Gilat. *MATLAB: An introduction with Applications.* John Wiley & Sons, 2004.

[17] Richard Lynn Huchzermeyer. *Measuring mechanical properties using digital image correlation: extracting tensile and fracture properties from a single sample.* PhD thesis, Stellenbosch: Stellenbosch University, 2017.

[18] Bruce D Lucas, Takeo Kanade, et al. An iterative image registration technique with an application to stereo vision. 1981.

[19] Q-T Luong and Olivier D Faugeras. Self-calibration of a moving camera from point correspondences and fundamental matrices. *International Journal of computer vision*, 22(3):261–289, 1997.

[20] JH Michell. On the direct determination of stress in an elastic solid, with application to the theory of plates. *Proceedings of the London Mathematical Society*, 1(1):100–124, 1899.

[21] Jean-José Orteu, Dorian Garcia, Laurent Robert, and Florian Bugarin. A speckle texture image generator. In *Speckle06: speckles, from grains to flowers*, volume 6341, page 63410H. International Society for Optics and Photonics, 2006.

[22] B Pan, K Li, and W Tong. Fast, robust and accurate digital image correlation calculation without redundant computations. *Experimental Mechanics*, 53(7):1277–1289, 2013.

[23] Bing Pan, Huimin Xie, Zhaoyang Wang, Kemao Qian, and Zhiyong Wang. Study on subset size selection in digital image correlation for speckle patterns. *Optics express*, 16(10):7037–7048, 2008.

[24] Zhiwei Pan, Wei Chen, Zhenyu Jiang, Liqun Tang, Yiping Liu, and Zejia Liu. Performance of global look-up table strategy in digital image correlation with cubic b-spline interpolation and bicubic interpolation. *Theoretical and Applied Mechanics Letters*, 6(3):126–130, 2016.

[25] J Poissant and F Barthelat. A novel subset splitting procedure for digital image correlation on discontinuous displacement fields. *Experimental mechanics*, 50(3):353–364, 2010.

[26] HI Richard. In defence of the 8-point algorithm. In *Proceedings of the 5th International Conference on Computer Vision*, pages 1064–1070. Cambridge: IEEE Computer Science Press, 1995.

[27] M.A. Sutton, J.J. Orteu, and H. Schreier. *Image Correlation for Shape, Motion and Deformation Measurements: Basic Concepts,Theory and Applications*. Springer US, 2009.

[28] Roger Tsai. A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses. *IEEE Journal on Robotics and Automation*, 3(4):323–344, 1987.

[29] Melody van Rooyen and Thorsten Hermann Becker. High-temperature tensile property measurements using digital image correlation over a non-uniform temperature field. *The Journal of Strain Analysis for Engineering Design*, 53(3):117–129, 2018.

[30] Guo-Qing Wei and Song De Ma. Implicit and explicit camera calibration: Theory and experiments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(5):469–480, 1994.

[31] ML Williams. On the stress distribution at the base of a stationary crack. 1997.

[32] JR Yates, M Zanganeh, D Asquith, and YH Tai. Quantifying crack tip displacement fields: Tstress and ctoa. In *CP2009*, 2009.

[33] Zhengyou Zhang. *Camera Calibration*. Prentice Hall PTR, 2004.

Appendices

Appendix A

Code for synthetic image generation

```
function [Fout,Gout]=analyticalspeckleContinuous7(stepsize,...
    specsize,fux,fuy,fillfactor,numpoints,lims)
    % This function is based on the methods used in the ...
        paper: 'Performance of sub-pixel registration ...
        algorithms in digital image correlation' using ...
        analytical formulas for speckles. And the code here ...
        has been adapted from code written by Matthew Molteno

    % create a grid of points which corresponding to the ...
        centroids of pixels
    x=lims(1):stepsize:lims(2);
    y=lims(4):-stepsize:lims(3);
    [X,Y]=meshgrid(x,y);
    [ysize,xsize]=size(X);
    % reshape X and Y to column vectors for the purpose of ...
        vectorisation (for GPU)
    X=reshape(X,[ysize*xsize,1]);
    Y=reshape(Y,[ysize*xsize,1]);

    % for super sampling create a grid of points that fall ...
        within the CCD array elements fillfactor area (xgrid1...
         and ygrid1)
    if numpoints>1
        % determine the stepsize for this grid of points
        xdist=(stepsize./2.*(1 - fillfactor./2) - ( - ...
            stepsize./2.*(1 - fillfactor./2)))/(numpoints-1);
        ydist=(stepsize./2.*(1 - fillfactor./2) - ( - ...
            stepsize./2.*(1 - fillfactor./2)))/(numpoints-1);
        % use the stepsize to define the grid of points
        xgrid1=( - stepsize./2.*(1 - fillfactor./2)):xdist:(...
             stepsize./2.*(1 - fillfactor./2));
        ygrid1=( - stepsize./2.*(1 - fillfactor./2)):ydist:(...
            stepsize./2.*(1 - fillfactor./2));
    else %if numpoints is 1 (no super sampling - to reduce ...
        computation time)
        xdist=0;
        ydist=0;
        xgrid1=0;
        ygrid1=0;
    end
    [xgrid2,ygrid2]=meshgrid(xgrid1,ygrid1);
    % jitter the grid of points so that the locations are ...
        somewhat random
    xgrid3=xgrid2 + (gpuArray.rand(numpoints,numpoints) - ...
        0.5).*xdist;
    ygrid3=ygrid2 + (gpuArray.rand(numpoints,numpoints) - ...
        0.5).*ydist;

    % reshape xgrid3 and ygrid3 to be row vectors and pass ...
        them to the GPU
    xxgrid=gpuArray(reshape(xgrid3,[1,numpoints*numpoints]))...
        ;
    yygrid=gpuArray(reshape(ygrid3,[1,numpoints*numpoints]))...
        ;

    % adjust specsize from units of pixels to the units used...
         by the displacement function
    specsize=(specsize(:)*(lims(2)-lims(1))/xsize)/2;
    % determine the speckle locations and size
    posx=rand(1,length(specsize))*(lims(2)-lims(1)) + lims...
        (1);
    posy=rand(1,length(specsize))*(lims(4)-lims(3)) + lims...
        (3);
    specamp=rand(length(specsize),1)/2+0.5; % speckle ...
        amplitude lies between 0.5 and 1

    % this function is used to adjust the amplitude of the ...
        sin function based on the number of repeated sine ...
        curves that fall within the range [0,2*pi] (...
        repetitions per revolution)
    T=@(x) 0.2275+2.1.*exp(-x);
    % this value adjusts the portion of the speckle radius ...
        which is dependent on the sine function
    amp=2

    % save speckle information in the form that it is needed
    for k=1:length(specsize)
        a(1,k)=rand([1,1]).*2.*pi;              % define a ...
            random phase shift
        b(1,k)=randi([1,5],[1,1]);              % define a ...
            random frequency of the sine function per ...
            revolution
        c(1,k)=(specsize(k)/amp).*(T(b(1,k)));  % define an ...
            amplitude of the sine function
        d(1,k)=specsize(k);                     % define the...
             base radius which the sine function changes
        Xk(1,k)=posx(k);                        % define the...
             x position of the speckle
        Yk(1,k)=posy(k);                        % define the...
             y position of the speckle
        I0(k,1)=specamp(k);                     % define the...
             light intensity at the centre of the speckle
    end

    % Determine how many elements can be processed at once ...
        on the GPU (due to ram limitations). The number ...
        44567000 is specific to the computer used during the ...
        project (4 GB of VRAM).
    div=(xsize.*ysize.*max(size(specsize))*numpoints*...
        numpoints)/(44567000);
    numOfPixels=floor(ysize*xsize/ceil(div));

    H=fspecial('average',[numpoints,numpoints]); % matrix to...
         average the light intensities for one pixel
    H=reshape(H,[1,numpoints,numpoints]); %reshape into a ...
        row vector
    HH=repmat(H,[numOfPixels,1,1]);  % create a larger ...
        matrix of repeated rows that is equivalent in size to...
         the number of elements to be generated

    Fout=zeros([ysize*xsize,1]);
    Gout=zeros([ysize*xsize,1]);

    for imageCount=1:2
        count=0;
        Xgrid=gpuArray.zeros([numOfPixels,numpoints,...
            numpoints]);
        Ygrid=Xgrid;
        for k=1:ceil(div)
            % determine the range of locations to save the ...
                pixel light intensity values to
            begin=1+count*numOfPixels;
            ending=(count+1)*numOfPixels;
            ending2=numOfPixels;
            % for the last iteration of k adjust the range ...
                of locations so that they do not go beyond ...
                the last pixel
            if ending>ysize*xsize
                ending=ysize*xsize;
                ending2=ysize*xsize-count*numOfPixels;
            end

            if imageCount==1 %if this is the reference image
                Xin=gpuArray(X(begin:ending))+xxgrid;
                Yin=gpuArray(Y(begin:ending))+yygrid;
            else % otherwise it is the deformed image (apply...
                 displacement function)
                Xin=gpuArray(X(begin:ending))-fux(X(begin...
                    :ending),Y(begin:ending))+xxgrid;
                Yin=gpuArray(Y(begin:ending))-fuy(X(begin...
                    :ending),Y(begin:ending))+yygrid;
            end

            [rrr1,rrr2]=size(Xin);
            % reshape the grid of points into a column ...
                vector for the purpose of matrix ...
                multiplication
            Xgrid=gpuArray(reshape(Xin,[rrr1*rrr2,1]));
            Ygrid=gpuArray(reshape(Yin,[rrr1*rrr2,1]));
            clear Xin Yin
            % determine the distance between the points ...
                under consideration and the speckle locations
            X1=Xgrid-Xk;
            Y1=Ygrid-Yk;
            clear Xgrid Ygrid
            % determine the angle from the centre of each ...
                speckle to each grid point to be evaluated
            angles=atan2(Y1,X1);
            % manipulate the angle by applying a predefined ...
                phase shift (b) and multiplying by c to ...
                change the frequency
            sinin=(angles+a).*b;
            clear angles
            % calculate the sin of the angles times its ...
                amplitude plus its shift to get the radius of...
                 the speckle in this specific direction
            sins1=sin(sinin).*c+d;
            clear sinin
            % square the radius of the speckle
            sins=sins1.^2;
            clear sins1
            % determine the distance squared between the ...
                speckle centre and the point under ...
                consideration
            X2=X1.^2;
            clear X1
            Y2=Y1.^2;
            clear Y1
            M=X2+Y2;
            clear X2 Y2
            % taking the exponent of the negative distance ...
                squared between the speckle centre and the ...
                point under consideration over the squared ...
                radius of the speckle. This function ensures ...
```

```
                that the light intensity for each speckle ...
                tapers off as points are analysed further and...
                 further from its centre
            speckleContribution=exp(-(M)./(sins));
            clear sins M
            Image_temp=speckleContribution*(I0);
            clear speckleContribution
            % reshape the points so that each row contains ...
                the supersampled light intensity values for ...
                each pixel
            Image_temp2=reshape(Image_temp,[rrr1,rrr2]);
            clear Image_temp
            % keep only the rows which are valid
            Image_sampled=Image_temp2(1:ending2,:);
            clear Image_temp2
            % use the guassian weighted average to determine...
                 the light intensity value for a pixel from ...
                its supersampled values
            Image_out_temp=sum(Image_sampled.*HH(1:ending2...
                ,:),2);
            clear Image_sampled
            % collect the pixel elements from the GPU and ...
                save them to the image matrix (on the GPU - ...
                in normal RAM)
            if imageCount==1
                Fout(begin:ending)=gather(Image_out_temp);
            else
                Gout(begin:ending)=gather(Image_out_temp);
            end
            clear Image_out_temp
            count=count+1;
        end
    end
    % reshape the column of pixels into a matrix
    Fout=reshape(Fout,[ysize,xsize]);
    Gout=reshape(Gout,[ysize,xsize]);
    % perform quantization on the light intensity values
    Fout=quantization(Fout,12);
    Gout=quantization(Gout,12);
end

function out=quantization(im,bit)
    % quantization to simulate the analogue to digital ...
        conversion of the CCD array
    num_levels=2^bit;
    stepsize=1/num_levels;
    bias=stepsize/2;
    values=0:stepsize:1;
    levels=values(2:end);
    out=imquantize(im,levels,values);
end
```

Appendix B

Displacement fields

B.1 Plate with hole

The displacement equations for a large plate containing a hole have been derived using the Airy stress function as done by Barber in [8]. The Airy stress function is a method of using equilibrium equations and boundary conditions of the specimen to solve for the two-dimensional stress and displacement fields experienced by the specimen.

For the case of a large plate with a circular hole in it experiencing a uniform normal stress, S, in the x direction the Airy stress function in terms of polar coordinates is

$$\phi = \frac{Sr^2}{4} - \frac{Sr^2\cos(2\theta)}{4} - \frac{Sa^2}{2} + \frac{Sa^2}{2}\cos(2\theta) - \frac{Sa^4}{4r^2}\cos(2\theta) \tag{B.1}$$

where a is the radius of the hole. The displacement equations are derived from this by substituting in the appropriate displacement components from the Michell solution [20]. After substitution the displacement equations become

$$\begin{aligned} f_r(r,\theta) = \frac{1}{2\mu}\Bigg\{ &\frac{S}{4}\left[(k-1)r\right] - \frac{S}{4}\left[-2r\cos(2\theta)\right] - \frac{Sa^2}{2}\left[-\frac{1}{r}\right] \\ &+ \frac{Sa^2}{2}\left[\frac{k+1}{r}\cos(2\theta)\right] - \frac{Sa^4}{4}\left[\frac{2}{r^3}\cos(2\theta)\right]\Bigg\} \end{aligned} \tag{B.2}$$

$$f_\theta(r,\theta) = \frac{1}{2\mu}\left\{-\frac{S}{4}\left[2r\sin(2\theta)\right] + \frac{Sa^2}{2}\left[-\frac{k-1}{r}\sin(2\theta)\right] - \frac{Sa^4}{4}\left[\frac{2}{r^3}\sin(2\theta)\right]\right\} \tag{B.3}$$

where the substituted displacement components are indicated by square brackets. Here f_r is radial displacement, f_θ is angular displacement, μ is the shear modulus and k is Kolosov's constant. These equations can be converted to describe the displacement in terms of cartesian coordinates using the following equations

$$f_u(r,\theta) = u_r(r,\theta)\cos(\theta) - u_\theta(r,\theta)\sin(\theta) \tag{B.4}$$

$$f_v(r,\theta) = u_r(r,\theta)\sin(\theta) + u_\theta(r,\theta)\cos(\theta) \tag{B.5}$$

$$\text{where} \quad r = \sqrt{x^2+y^2} \quad \text{and} \quad \theta = \operatorname{atan2}(y,x). \tag{B.6}$$

Note that atan2 is a Matlab function which computes the arctan of a vector and returns the angle corresponding to the correct quadrant within which that vector lies. The displacement equations in cartesian coordinates become

$$\begin{aligned} u_x(x,y,S) = \frac{S\,a}{8\,\mu}\Bigg[&\frac{2\,a}{\sqrt{x^2+y^2}}\left(\cos\left(3\operatorname{atan2}(y,x)\right) + \cos\left(\operatorname{atan2}(y,x)\right)\,(k+1)\right) \\ &- \frac{2\,a^3}{\left(\sqrt{x^2+y^2}\right)^3}\cos\left(3\operatorname{atan2}(y,x)\right) + \frac{\sqrt{x^2+y^2}}{a}\cos\left(\operatorname{atan2}(y,x)\right)\,(k+1)\Bigg] \end{aligned} \tag{B.7}$$

$$\begin{aligned} u_y(x,y,S) = \frac{S\,a}{8\,\mu}\Bigg[&\frac{2\,a}{\sqrt{x^2+y^2}}\left(\sin\left(3\operatorname{atan2}(y,x)\right) - \sin\left(\operatorname{atan2}(y,x)\right)\,(k-1)\right) \\ &- \frac{2\,a^3}{\left(\sqrt{x^2+y^2}\right)^3}\sin\left(3\operatorname{atan2}(y,x)\right) + \frac{\sqrt{x^2+y^2}}{a}\sin\left(\operatorname{atan2}(y,x)\right)\,(k-3)\Bigg] \end{aligned} \tag{B.8}$$

where the stress, S, has been made a variable of the function.

B.2 Plate with crack

These equations for Cartesian displacements in terms of polar coordinates, for mode I and II crack propagation, are of the form [32]

Mode I:

$$f_u(r,\theta) = \sum_{n=1}^{\infty} \frac{r^{\frac{n}{2}}}{2\mu} a_n \left[\left(k + \frac{n}{2} + (-1)^n\right)\cos\frac{n\theta}{2} - \frac{n}{2}\cos\frac{(n-4)\theta}{2}\right] \tag{B.9}$$

$$f_v(r,\theta) = \sum_{n=1}^{\infty} \frac{r^{\frac{n}{2}}}{2\mu} a_n \left[\left(k - \frac{n}{2} - (-1)^n\right)\sin\frac{n\theta}{2} + \frac{n}{2}\sin\frac{(n-4)\theta}{2}\right] \tag{B.10}$$

Mode II:

$$f_u(r,\theta) = -\sum_{n=1}^{\infty} \frac{r^{\frac{n}{2}}}{2\mu} b_n \left[\left(k + \frac{n}{2} - (-1)^n\right)\sin\frac{n\theta}{2} - \frac{n}{2}\cos\frac{(n-4)\theta}{2}\right] \tag{B.11}$$

$$f_v(r,\theta) = \sum_{n=1}^{\infty} \frac{r^{\frac{n}{2}}}{2\mu} b_n \left[\left(k - \frac{n}{2} + (-1)^n \right) \cos \frac{n\theta}{2} + \frac{n}{2} \cos \frac{(n-4)\theta}{2} \right] \tag{B.12}$$

Using Equation B.6 these equations can be rewritten in purely Cartesian coordinates as

Mode I:

$$f_u(x,y) = \sum_{n=1}^{\infty} \frac{(x^2+y^2)^{\frac{n}{4}}}{2\mu} a_n \left[\left(k + \frac{n}{2} + (-1)^n \right) \cos \frac{n.\mathrm{atan2}(y,x)}{2} - \frac{n}{2} \cos \frac{(n-4)\mathrm{atan2}(y,x)}{2} \right] \tag{B.13}$$

$$f_v(x,y) = \sum_{n=1}^{\infty} \frac{(x^2+y^2)^{\frac{n}{4}}}{2\mu} a_n \left[\left(k - \frac{n}{2} - (-1)^n \right) \sin \frac{n.\mathrm{atan2}(y,x)}{2} + \frac{n}{2} \sin \frac{(n-4)\mathrm{atan2}(y,x)}{2} \right] \tag{B.14}$$

Mode II:

$$f_u(x,y) = -\sum_{n=1}^{\infty} \frac{(x^2+y^2)^{\frac{n}{4}}}{2\mu} b_n \left[\left(k + \frac{n}{2} - (-1)^n \right) \sin \frac{n.\mathrm{atan2}(y,x)}{2} - \frac{n}{2} \cos \frac{(n-4)\mathrm{atan2}(y,x)}{2} \right] \tag{B.15}$$

$$f_v(x,y) = \sum_{n=1}^{\infty} \frac{(x^2+y^2)^{\frac{n}{4}}}{2\mu} b_n \left[\left(k - \frac{n}{2} + (-1)^n \right) \cos \frac{n.\mathrm{atan2}(y,x)}{2} + \frac{n}{2} \cos \frac{(n-4)\mathrm{atan2}(y,x)}{2} \right] \tag{B.16}$$